DIAGRAMMES

C. ORIAT

Étude des spécifications modulaires : constructions de colimites finies, diagrammes, isomorphismes

Diagrammes, tome 39 (1998), p. 3-120

<http://www.numdam.org/item?id=DIA_1998__39__3_0>

© Université Paris 7, UER math., 1998, tous droits réservés.

L'accès aux archives de la revue « Diagrammes » implique l'accord avec les conditions générales d'utilisation (http://www.numdam.org/legal.php). Toute utilisation commerciale ou impression systématique est constitutive d'une infraction pénale. Toute copie ou impression de ce fichier doit contenir la présente mention de copyright.

NUMDAM

Article numérisé dans le cadre du programme
Numérisation de documents anciens mathématiques
http://www.numdam.org/

DIAGRAMMES VOLUME 39 , 1998

ETUDE DES SPECIFICATIONS MODULAIRES : CONSTRUCTIONS DE COLIMITES FINIES, DIAGRAMMES, ISOMORPHISMES (*)

PARTIE I (**)

C. Oriat

RESUME. La composition de spécifications modulaires peut être modélisée, dans le formalisme des catégories, par des colimites de diagrammes. La somme amalgamée permet en particulier d'assembler deux spécifications en précisant les parties communes. Notre travail poursuit cette idée classique selon trois axes.

D'un point de vue syntaxique, nous définissons un langage pour représenter les spécifications modulaires construites à partir d'une catégorie de spécifications et de morphismes de spécifications de base. Ce langage est caractérisé formellement par une catégorie de termes finiment cocomplète.

D'un point de vue sémantique, nous proposons d'associer à tout terme un diagramme. Cette interprétation permet de faire abstraction de certains choix effectués lors de la construction de la spécification modulaire. Pour cela, nous définissons une catégorie de diagrammes "concrète", c'est-à-dire dont les flèches peuvent être manipulées effectivement. En considérant le quotient par une certaine congruence, nous obtenons une complétion de la catégorie de base par colimites finies. Nous montrons que le calcul du diagramme associé à un terme définit une équivalence entre la catégorie des termes et la catégorie des diagrammes, ce qui prouve la correction de cette interprétation.

Enfin, nous proposons un algorithme pour décider si deux diagrammes sont isomorphes, dans le cas particulier où la catégorie de base est finie et sans cycle. Cela permet de détecter des isomorphismes "de construction" entre spécifications modulaires, c'est-à-dire des isomorphismes qui ne dépendent pas des spécifications de base, mais seulement de la manière dont celles-ci sont assemblées.

ABSTRACT. The composition of modular specifications can be modeled, in a categoy theoretic framework, by means of colimits of diagrams. Pushouts in particular allow us to

---

(*) Thèse d'Informatique, préparée et soutenue à l'Institut National Polytechnique de Grenoble (Laboratoire LSR-IMAG).

(**) La Partie II de ce travail sera publiée dans Diagrammes 40 et la Partie III dans Diagrammes 41.

gather two specifications sharing a common part. Our work extends this classic idea along three lines.

From a syntactic point of view, we define a language to represent modular specifications built from a category of base specifications and base specification morphisms. This language is formally characterized by a finitely cocomplete category of terms.

From a semantic point of view, we propose to associate with each term a diagram. This interpretation allows us to abstract some choices made while constructing a modular specification. We thus define a "concrete" category of diagrams, in which arrows can actually be handled. Considering the quotient by a certain congruence relation, we get a completion of the base category with finite colimits. We prove that this calculus defines an equivalence between the category of terms and the category of diagrams, which shows the soundness of this interpretation.

At last, we propose an algorithm to decide whether two diagrams are isomorphic, when the base category is finite and cycle free. This allows us to detect "construction isomorphisms" between modular specifications, i.e. isomorphisms which do not depend on the base specifications, but only on their combination.

Remerciements

Je remercie tout d'abord mon directeur de thèse, M. Didier Bert, pour son soutien et pour la confiance qu'il m'a accordée dans la préparation de cette thèse.

Je remercie M. Jean-Pierre Verjus qui a accepté de présider ce jury.

Je remercie M. Michel Bidoit pour l'intérêt qu'il a porté à mon travail et pour avoir accepté d'être rapporteur de ma thèse.

Je remercie M. Christian Lair pour les améliorations qu'il m'a suggérées et pour les erreurs qu'il m'a signalées avec beaucoup d'indulgence et de gentillesse. Je suis particulièrement honorée qu'un catégoricien de sa compétence ait accepté de faire partie de mon jury en qualité de rapporteur.

Je remercie M. Michel Sintzoff pour m'avoir fait l'honneur de participer à ce jury.

Ce travail doit beaucoup à M. Jean-Claude Reynaud. Je le remercie d'une part pour le temps qu'il a consacré à suivre mon travail, et d'autre part pour toutes les suggestions qu'il m'a faites. Si, par hasard, ce travail contient quelques idées intéressantes, c'est probablement lui qui me les a soufflées (mais, en paraphrasant Mac Lane, je suis évidemment responsable des erreurs qui pourraient rester).

Je remercie enfin les membres de l'équipe SCOP, auxquels j'associe les occupants présents ou passés du bâtiment *D*, pour leur accueil et leur soutien tout au long de ce travail.

"Le département [...] épousait l'opinion traditionnelle qui voulait que la recherche fût une occupation solitaire et érémitique, une mise à l'épreuve du caractère plus que de l'érudition, qu'un abus de contact humain pouvait vicier."
— David Lodge, La chute du British Museum

Introduction

Spécifier un problème consiste à le décrire sans décider prématurément de la façon dont il sera résolu. En informatique, une spécification de programme est une description des fonctionnalités attendues du programme indépendante des choix de mise en œuvre.

Il existe plusieurs manières de spécifier un programme. On peut utiliser une langue naturelle, par exemple en écrivant un cahier des charges. Certaines activités, comme la validation de programme, la construction automatique d'un prototype, ou la production systématique de jeux de tests, nécessitent des spécifications *formelles*, qui possèdent une sémantique précise, en particulier non ambiguë.

Nous nous intéressons ici aux *spécifications algébriques*, qui font partie des spécifications *orientées propriétés*[1]. Les spécifications orientées propriétés sont basées sur un *système logique*, qui définit l'ensemble des propriétés que doit satisfaire tout modèle de la spécification.

1 Spécifications algébriques

Une spécification algébrique est composée d'une signature et d'un ensemble d'axiomes. La signature contient les symboles utilisés pour former des expressions, et les axiomes définissent certaines propriétés de ces expressions. La logique sous-jacente décrit l'ensemble des *théorèmes* que l'on peut déduire des axiomes. De plus, la *sémantique* des spécifications algébriques associe à toute spécification une classe de *modèles*, c'est-à-dire une classe d'algèbres multi-sortes qui satisfont les axiomes.

Historiquement, les spécifications algébriques sont issues de deux courants distincts : l'algèbre universelle en mathématiques [Coh65] et les types abstraits en génie logiciel. L'origine des types abstraits figure dans le concept de *classe* du langage de programmation Simula [DN66, BDMN73]. Un type abstrait, composé d'un ensemble de *domaines de valeurs* accompagné d'un ensemble d'*opérations* sur ces domaines de valeurs, permet de structurer les données. L'idée de base en spécification algébrique consiste à modéliser un type abstrait par une algèbre. Cette idée est motivée par l'analogie entre un type abstrait et une algèbre multi-sortes sur une signature. L'interprétation des sortes de la signature correspond en effet aux domaines de valeurs, et l'interprétation des opérateurs de la signature correspond aux opérations sur ces domaines de valeurs.

Les premiers travaux sur les spécifications algébriques datent du milieu des années 1970, avec les travaux de B. Liskov et S. Zilles [LZ74], de J. Guttag [Gut75,

1. En anglais, *property-oriented.*

Gut77] et du groupe ADJ (composé de J. Goguen, J. Thatcher, E. Wagner et J. Wright) [GTWW75, GTWW77, GTW78]. Au départ, les spécifications algébriques étaient basées sur la logique équationnelle. Puis ont été introduites d'autres logiques, comme les clauses de Horn, la logique du premier ordre ou encore les spécifications avec contraintes [EM90]. D'autres extensions ont permis de prendre en compte la notion d'erreur [BBC86, BL93]. J. Goguen et R. Burstall ont proposé la *théorie des institutions*, qui offre un cadre général pour étudier les différents formalismes de spécification algébrique [GB84, GB90]. Il existe beaucoup de références sur les spécifications algébriques, parmi les bonnes synthèses, citons [EM85, MG85, Wir90].

Ces travaux ont donné naissance à de nombreux langages de spécification algébrique, comme Clear [BG77, BG80], ACT ONE et ACT TWO [EM85, EM90], ASL [SW83, Wir86], OBJ2 et OBJ3 [FGJM85, GKK+87], PLUSS [Gau84, Bid89], LPG [Ber83, BE86, B+90], GLIDER [Huf92]. On trouve un bon panorama des différents langages de spécification algébrique existants dans [Wir94].

2 Théorie des catégories

La théorie des catégories repose sur deux notions : l'*objet* et la *flèche*, contrairement à la théorie des ensembles qui repose sur le seul concept d'*ensemble*. En théorie des ensembles, un ensemble est caractérisé de façon *interne* par ses *éléments*, alors qu'en théorie des catégories, un *objet* est caractérisé de façon *externe* par les relations entretenues par l'intermédiaire des flèches avec les autres objets. Pour cette raison, un grand nombre de concepts de théorie des catégories sont définis *à un isomorphisme près*.

La théorie des catégories a été inventée au début des années 1940 par S. Mac Lane et S. Eilenberg. Bien que certains mathématiciens aient douté de l'intérêt des catégories, ce formalisme a joué un rôle unificateur en mathématiques, en particulier en algèbre et en topologie. Cette théorie définit les différents concepts de façon générale et abstraite, indépendamment de tout modèle. (C'est pour cette raison que ses détracteurs l'on surnommée "non-sens abstrait[2]".) La théorie des catégories a commencé à intéresser les informaticiens au début des années 1970. Par exemple, la découverte de l'équivalence entre le λ-calcul typé et les catégories cartésiennes fermées a permis de donner des solutions élégantes au problème des modèles du λ-calcul, lié à celui de la résolution des équations de domaines [SP77, Wan79, LS86]. Il existe de façon générale des liens très étroits entre la théorie des types et la théorie des catégories (cf. par exemple [See84, Poi92]).

L'ouvrage de référence sur les catégories est le livre de S. Mac Lane [McL71], plutôt destiné aux mathématiciens. Pour les informaticiens, le livre de M. Barr et C. Wells [BW90] est plus accessible, car les différents concepts sont présentés de façon assez élémentaire et les exemples sont tirés de l'informatique. Il existe beaucoup d'autres ouvrages qui traitent des catégories, par exemple [MB70, AM75, Gol79, AL91].

Le formalisme des spécifications algébriques s'appuie largement sur la théorie

2. En anglais, *abstract nonsense*.

des catégories. Historiquement, le développement des spécifications algébriques (en particulier les travaux du groupe ADJ) a été influencé par les théories algébriques de Lawvere. Les théories algébriques de Lawvere permettent de modéliser les catégories d'algèbres en faisant abstraction de la syntaxe, c'est-à-dire indépendamment de la notion de signature [Law63, BW85, WBT85].

D'autre part, d'un point de vue pratique, on peut justifier l'utilisation des catégories par l'importance des *morphismes de spécifications*, qui jouent un rôle essentiel en spécification algébrique. Intuitivement, un morphisme de spécifications exprime une relation entre deux spécifications. Pour prendre un exemple mathématique, tout anneau peut être considéré comme un groupe, ce qui signifie qu'il existe un morphisme de spécifications entre la spécification des groupes et la spécification des anneaux. Il peut exister plusieurs morphismes de spécifications entre deux spécifications. Par exemple, on peut considérer tout anneau comme un monoïde *de deux façons différentes*, en considérant soit l'opérateur additif, soit l'opérateur multiplicatif de l'anneau comme l'opérateur du monoïde. Préciser quel opérateur de l'anneau sera l'opérateur du monoïde consiste exactement à définir un morphisme de spécifications entre la spécification des monoïdes et la spécification des anneaux.

3 Modularité

Le développement de spécifications de grande taille nécessite un découpage des spécifications en plusieurs spécifications plus simples, appelées *modules*. Chaque module correspond à un problème plus simple à résoudre. Cette méthode, qui consiste à "diviser pour mieux régner" est classique en génie logiciel [LCW85]. De façon générale, la modularité permet d'une part le développement indépendant des différents modules, et d'autre part une meilleure compréhension de chaque module, ce qui facilite la maintenance.

Cette vision de la modularité correspond à une approche descendante de résolution de problème, puisque l'accent est mis sur la *décomposition* du problème en plusieurs sous-problèmes plus simples. Dans notre travail, notre conception de la modularité est au contraire ascendante. Nous nous intéressons en effet à la *composition* des modules, c'est-à-dire à l'assemblage de modules élémentaires permettant de construire des modules plus complexes. Ces modules élémentaires sont regroupés dans une bibliothèque, la bibliothèque des *modules de base*. L'intérêt de la composition est de permettre la *réutilisation* des modules de base, préconisée en génie logiciel pour réduire les erreurs et les coûts de développement (cf. par exemple [Ber90]).

Un aspect de la modularité est la possibilité de définir des spécifications *génériques*, c'est-à-dire paramétrées [SST92]. En effet, une spécification paramétrée peut être instanciée de différentes manières par d'autres spécifications, ce qui évite de réécrire une nouvelle spécification pour chaque variation d'un même problème. Le but est alors de définir des spécifications génériques suffisamment générales pour permettre de spécifier le maximum de problèmes par une simple instanciation [Mar95].

Le développement modulaire des spécifications est assisté dans les langages de spécification algébrique par des *opérateurs de construction*[3] comme le *renommage*,

3. En anglais, *specification-building operations*.

l'*enrichissement* ou l'*abstraction* [ST88, Wir90]. Ces primitives permettent de construire de nouvelles spécifications à partir de spécifications déjà définies. La composition de modules est un cas particulier d'opérateur de construction de spécifications, qui peut être modélisé par des *colimites de diagrammes*. Intuitivement, en théorie des catégories, un *diagramme* permet de décrire un assemblage de plusieurs objets en précisant des *partages* entre certains objets à l'aide de flèches. La *colimite* du diagramme permet de considérer le résultat de cette composition. Cette utilisation générale des colimites pour modéliser l'interconnexion de systèmes a été proposée par J. Goguen [Gog73, Gog92].

Un cas particulier de colimite est la *somme amalgamée*[4], qui correspond à l'assemblage de deux objets B et C, en spécifiant une partie commune entre ces deux objets par un *objet partagé* A et deux flèches $f : A \to B$ et $g : A \to C$. En spécification algébrique, la somme amalgamée peut modéliser d'une part la *composition* de deux spécifications qui ont une partie commune, et d'autre part l'*instanciation* d'une spécification générique.

4 Notre travail

Nous étudions la composition des spécifications algébriques modulaires. Nous supposons que nous disposons d'une bibliothèque de spécifications et de morphismes de spécifications *de base*, qui forment une catégorie *de base*, en général appelée $\mathcal{C}_0$ dans ce mémoire. Les spécifications de base sont monolithiques, c'est-à-dire non décomposées. Notre travail repose sur une idée classique en spécification algébrique : la composition des spécifications peut être modélisée par des colimites de diagrammes finis. Nous proposons donc de considérer une *spécification modulaire* comme une spécification obtenue à partir de spécifications et morphismes de spécifications de base par une suite de constructions de colimites.

Notre travail est indépendant de la logique sous-jacente du langage de spécification algébrique. Nous pouvons donc nous placer dans le cadre général des *institutions*. Nous devons uniquement supposer que la catégorie des spécifications est *finiment cocomplète*, c'est-à-dire que tout diagramme fini a une colimite.

D'un point de vue syntaxique, c'est-à-dire en ce qui concerne le *langage* utilisé pour décrire ces constructions, nous ne considérons pas toutes les colimites finies, mais uniquement les sommes amalgamées. En effet, avec la spécification vide, objet initial de la catégorie des spécifications, les sommes amalgamées permettent de simuler la construction de toute colimite finie. Nous proposons la construction d'une catégorie de *termes*, appelée Terme($\mathcal{C}_0$), pour représenter la composition de spécifications modulaires en utilisant les sommes amalgamées.

En théorie des catégories, on travaille le plus souvent à un isomorphisme près. C'est une conséquence de la caractérisation externe des objets dans ce formalisme. Les colimites, par exemple, sont définies au départ à un isomorphisme près. La définition d'une syntaxe est au contraire un problème typiquement informatique, qui nécessite de faire des *choix de représentation*. C'est la raison pour laquelle nous devons utiliser des *sommes amalgamées choisies* au moment de la définition du langage

4. En anglais, *pushout*. Les catégoriciens utilisent également le terme de *coproduit fibré*.

d'expression pour les spécifications modulaires. Ces choix de sommes amalgamées correspondent à des choix de codage.

Un terme correspond donc à une spécification modulaire construite par une succession de sommes amalgamées. Dans une seconde étape, nous proposons d'interpréter un terme par un *diagramme* dont la colimite est la spécification modulaire dénotées par ce terme. Nous considérons les diagrammes comme une sémantique pour les spécifications modulaires, par opposition aux termes qui constituent la syntaxe.

L'interprétation des termes par des diagrammes nécessite de définir une *catégorie* de diagrammes puisqu'il faut non seulement associer à toute spécification modulaire un diagramme, mais aussi associer à tout morphisme de spécifications modulaires un morphisme de diagrammes. La définition d'un diagramme est relativement standard en théorie des catégories, bien qu'il existe quelques nuances. La définition de morphisme de diagrammes est moins connue, en particulier dans la littérature informatique où sont utilisées des définitions moins générales que celle dont nous nous servons ici.

Nous présentons deux catégories de diagrammes, DIAGR($\mathcal{C}_0$) et diagr($\mathcal{C}_0$). Les objets de DIAGR($\mathcal{C}_0$) sont des *diagrammes*, et les flèches de DIAGR($\mathcal{C}_0$) des *morphismes de diagrammes*. La catégorie diagr($\mathcal{C}_0$) est obtenue à partir de DIAGR($\mathcal{C}_0$) par un quotient par une relation de congruence sur les flèches de DIAGR($\mathcal{C}_0$). C'est la catégorie diagr($\mathcal{C}_0$) qui possède les propriétés théoriques intéressantes. Il s'agit en effet d'une catégorie *finiment cocomplète*, c'est-à-dire que tout diagramme fini sur diagr($\mathcal{C}_0$) a une colimite finie dans diagr($\mathcal{C}_0$). D'autre part, diagr($\mathcal{C}_0$) est une complétion par colimites finies de la catégorie $\mathcal{C}_0$. Sur le plan pratique, comme les flèches de diagr($\mathcal{C}_0$) sont des classes d'équivalence de morphismes de diagrammes, on travaille en réalité avec des représentants, c'est-à-dire avec des flèches de DIAGR($\mathcal{C}_0$). La catégorie DIAGR($\mathcal{C}_0$) permet donc la manipulation effective des flèches de diagr($\mathcal{C}_0$).

La représentation des termes par des diagrammes est définie par un foncteur

$$\mathcal{D} : \text{Terme}(\mathcal{C}_0) \to \text{diagr}(\mathcal{C}_0)$$

entre les catégories Terme($\mathcal{C}_0$) et diagr($\mathcal{C}_0$). Nous montrons que ce foncteur définit une *équivalence de catégories* entre Terme($\mathcal{C}_0$) et diagr($\mathcal{C}_0$), ce qui signifie que bien qu'elles ne soient pas isomorphes, ces catégories ont néanmoins essentiellement la même structure.

Après avoir représenté les spécifications modulaires par des diagrammes, nous nous intéressons au problème de l'*équivalence* entre deux spécifications modulaires. Deux spécifications modulaires sont considérées comme équivalentes lorsque celles-ci sont *isomorphes* en tant que *colimites de diagramme*. Détecter cet isomorphisme consiste en fait à détecter un isomorphisme entre deux diagrammes dans la catégorie diagr($\mathcal{C}_0$). Nous appelons cet isomorphisme un *isomorphisme de construction*, parce qu'il ne dépend pas de la définition effective des spécifications de base, mais seulement de la manière dont celles-ci sont combinées pour construire les spécifications modulaires. Dans l'hypothèse où la catégorie $\mathcal{C}_0$ est finie et ne comporte pas de cycle, nous proposons un algorithme pour détecter si deux diagrammes sont isomorphes. Des résultats partiels concernant l'interprétation des termes par des diagrammes et la détection d'isomorphismes de construction ont été présentés dans [Ori94, Ori95].

5 Comparaisons avec d'autres travaux

Catégories de diagrammes

La définition de *diagramme* est, à quelques nuances près, standard. Chez S. Mac Lane [McL71], un diagramme sur une catégorie $\mathcal{C}$ est un *foncteur* d'une catégorie J vers $\mathcal{C}$. La définition de M. Barr et C. Wells [BW90], est moins générale, puisqu'un diagramme est un *morphisme de graphes* d'un graphe vers le graphe sous-jacent de $\mathcal{C}$. Nous utilisons une définition "intermédiaire" : pour nous, un diagramme est un *foncteur* d'une catégorie *librement engendrée sur un graphe* vers $\mathcal{C}$. Partir d'un graphe nous permet de rester proche de l'informatique. Par contre, nous ne souhaitons pas nous restreindre à une *petite* catégorie $\mathcal{C}$.

La définition de *morphisme de diagrammes* est beaucoup moins standard. Par exemple dans Clear, un morphisme de diagrammes, appelé *morphisme basé*, correspond uniquement à une *inclusion* entre deux diagrammes.

La définition proposée par A. Tarlecki, R. Burstall et J. Goguen dans [TBG91], bien que plus générale que celle utilisée dans la sémantique du langage Clear, n'est pas encore suffisamment générale pour notre travail. En effet, cette définition ne permet pas de représenter tout morphisme de spécifications modulaires par un morphisme de diagrammes.

La définition de morphisme de diagrammes que nous utilisons n'est pas nouvelle en théorie des catégories. Une version duale de la catégorie $\mathrm{diagr}(\mathcal{C}_0)$ est par exemple utilisée par M. Barr et C. Wells dans [BW94]. Néanmoins, la reformulation de la définition de catégorie de diagrammes que nous proposons dans le chapitre 2 présente un intérêt en informatique, d'une part parce que la catégorie $\mathrm{DIAGR}(\mathcal{C}_0)$ permet de manipuler effectivement les morphismes de diagrammes, et d'autre part parce que les catégories $\mathrm{DIAGR}(\mathcal{C}_0)$ et $\mathrm{diagr}(\mathcal{C}_0)$ sont peu connues dans cette discipline.

Calculs de colimites

R. Burstall et D. Rydeheard proposent des algorithmes pour *calculer* certains concepts de théorie des catégories [Bur80, BR86], en particulier les colimites de diagrammes. Le calcul général de colimite est paramétré par certaines colimites, en l'occurrence un objet initial, des sommes et des coégalisateurs, qui permettent d'évaluer la colimite d'un diagramme quelconque. Ainsi, ayant défini l'objet initial, la somme et le coégalisateur par exemple dans la catégorie des ensembles **Set**, on en déduit un moyen de calculer la colimite d'un diagramme quelconque dans **Set**. Cet algorithme est en fait basé sur une preuve du résultat suivant : si une catégorie a un objet initial, des sommes et des coégalisateurs, alors celle-ci est finiment cocomplète. Notre travail est basé sur un théorème similaire : si une catégorie a un objet initial et des sommes amalgamées alors celle-ci est finiment cocomplète. Notre travail est différent, dans la mesure où nous ne cherchons pas à calculer des colimites dans une catégorie fixée. En particulier, nous nous intéressons au problème général de l'isomorphisme entre deux diagrammes, qui correspond à un isomorphisme entre leurs colimites respectives. Cet isomorphisme est indépendant de la catégorie sur laquelle ces diagrammes sont construits.

Utilisation des colimites en spécification algébrique

L'utilisation des colimites pour représenter l'assemblage de spécifications est loin d'être nouvelle. R. Burstall et J. Goguen ont présenté cette idée dans la sémantique du langage de spécification Clear [BG80]. Dans Clear, la composition de plusieurs spécifications qui partagent un environnement commun, appelées *théories basées*, est en effet formalisée par la colimite d'un diagramme.

Les colimites ont ensuite été assez intensivement utilisées en spécification algébrique. La somme amalgamée en particulier permet de modéliser l'instanciation de spécifications génériques [TWW82, EM85]. H. Ehrig, R. Jimenez et F. Orejas [EJO93] ont également présenté une forme plus générale d'instanciation à l'aide de *sommes amalgamées multiples*[5], nouvel exemple de colimite.

Syntaxe pour les constructions modulaires

J. Bergstra, J. Heering et R. Klint [BHK90] d'une part, et G. Renardel de Lavalette [Ren91] d'autre part, ont proposé de structurer les opérateurs de construction de spécifications sous forme d'*algèbres de modules*. Une algèbre de modules est un *langage* permettant d'écrire des *expressions* de composition des spécifications algébriques. Cette approche est voisine de notre travail sur le langage Terme($\mathcal{C}_0$), puisque le problème de la composition des spécifications algébriques est abordé d'un point de vue *syntaxique*, et permet de comparer différentes spécifications modulaires. Par contre, nous ne nous intéressons pas aux mêmes opérateurs de constructions de spécifications, puisque nous ne considérons que les constructions de colimites.

Notre syntaxe pour représenter les spécifications algébriques modulaires est largement inspirée des travaux de J.-C. Reynaud, qui a proposé un système de types pour représenter les constructions de colimites [Rey90a, Rey90b, Rey93]. Ce système est fondé sur les théories algébriques généralisées de Cartmell [Car86] qui permettent de spécifier des types dépendants, c'est-à-dire des types paramétrés par des termes. Les types dépendants ont été également utilisés par T. Streicher et M. Wirsing [SW91] pour modéliser la composition de spécifications algébriques modulaires. D'autre part, H. Ehrig, R. Jimenez et F. Orejas ont également proposé une syntaxe pour les spécifications algébriques modulaires dans [EJO93]. Cette syntaxe est en partie basée sur des constructions de sommes amalgamées.

La principale difficulté en ce qui concerne la définition d'une syntaxe pour les constructions de colimites est le problème de circularité entre la définition d'une part des objets et des flèches de Terme($\mathcal{C}_0$) et d'autre part de l'égalité entre deux flèches de Terme($\mathcal{C}_0$). En effet, il existe une flèche, que nous appelons **up** dans ce mémoire, entre une somme amalgamée et un objet *à condition qu'une égalité entre deux flèches de* Terme($\mathcal{C}_0$) *soit vérifiée*. La définition des flèches **up** dépend donc de la définition de l'égalité des flèches dans Terme($\mathcal{C}_0$), qui elle-même présuppose que les flèches ont été définies.

J.-C. Reynaud contourne le problème en proposant une définition *concrète*, c'est-à-dire sémantique, d'une catégorie librement engendrée par objet initial choisi et sommes amalgamées choisies [Rey93]. H. Ehrig, R. Jimenez et F. Orejas [EJO93] ne

5. En anglais, *multiple pushout*.

définissent pas de syntaxe pour les flèches **up**. La syntaxe qu'ils proposent ne permet donc pas d'obtenir une catégorie *finiment cocomplète*.

Une solution pour spécifier une catégorie finiment cocomplète sans utiliser d'égalités pour définir les flèches a été présentée par F. Cury dans [Cur91]. F. Cury propose de dédoubler les flèches **up** en deux flèches dont la définition ne dépend pas d'une égalité. Ces deux flèches, qui sont des exemples particuliers de flèches **up**, permettent de reconstruire *a posteriori* toutes les flèches **up**.

La solution que nous proposons consiste à stratifier la construction de Terme($\mathcal{C}_0$) en une suite de catégories $\mathcal{C}_i$. L'existence d'une flèche **up** dans la catégorie $\mathcal{C}_i$ dépend uniquement d'égalités dans la catégorie $\mathcal{C}_{i-1}$. Nous évitons ainsi le problème de circularité entre la définition des flèches et la définition des égalités.

Notre catégorie Terme($\mathcal{C}_0$) peut être comparée au *type* d'une esquisse, introduit par C. Ehresmann [Ehr68]. Dans les travaux de D. Duval et J.-C. Reynaud [DR94a, DR94b], le type d'une esquisse est une *catégorie librement engendrée* à partir d'un graphe par *sommes* et *produits*. Par conséquent, la construction du type d'une esquisse est basée sur les catégories à sommes et produits alors que la construction de Terme($\mathcal{C}_0$) est basée sur les catégories à colimites finies. D'autre part, une esquisse contient des cônes et cocônes distingués, qui spécifient certaines limites ou colimites. Par contre, dans la construction de Terme($\mathcal{C}_0$) que nous proposons, il n'est pas possible de spécifier des sommes amalgamées distinguées dans la catégorie de base $\mathcal{C}_0$.

6 Plan de ce mémoire

Ce mémoire est divisé en trois parties, chaque partie faisant l'objet d'un volume de *Diagrammes*.

Ce volume contient la première partie de notre thèse, c'est-à-dire les chapitres 1 et 2, consacrés respectivement au cadre général (les spécifications algébriques modulaires) et aux bases théoriques (les catégories de diagrammes).

Le chapitre 1 introduit les spécifications algébriques modulaires, présente le problème de la composition des spécifications et pose le cadre général de notre travail. Après quelques rappels sur les spécifications algébriques, nous proposons plusieurs exemples de spécifications modulaires d'*anneaux*, en utilisant quelques spécifications et morphismes de spécifications de base. Cet exemple académique présente d'une part la syntaxe pour les constructions modulaires, et d'autre part le problème de l'isomorphisme de construction entre deux spécifications modulaires.

Le chapitre 2 contient les bases théoriques de notre travail. Nous effectuons quelques rappels de théorie des graphes en section 2.1, puis de théorie des catégories en section 2.2. Enfin, en section 2.3, nous présentons les notions de diagrammes, de colimites finies ainsi que les catégories de diagrammes DIAGR($\mathcal{C}_0$) et diagr($\mathcal{C}_0$). Nous montrons que la catégorie diagr($\mathcal{C}_0$) est finiment cocomplète, et que diagr($\mathcal{C}_0$) est une complétion par colimites finies de $\mathcal{C}_0$.

La deuxième partie, qui correspond au chapitre 3, est consacrée à la définition de la syntaxe des spécifications modulaires.

Enfin, dans la troisième partie, nous proposons d'interpréter les termes par des diagrammes (chapitre 4), et nous présentons un algorithme pour détecter si deux

diagrammes sont isomorphes, dans le cas particulier ou la catégorie de base $\mathcal{C}_0$ est finie et ne comporte pas de cycle (chapitre 5).

Chapitre 1

Motivation

Le but de ce chapitre est de présenter le problème de la composition des spécifications modulaires. Nous étudions différentes spécifications modulaires des anneaux, construites à partir de quelques spécifications de base. Cet exemple est décrit en utilisant le langage de spécification LPG (Langage pour la Programmation Générique) développé par D. Bert et R. Echahed [BE86, B+90]. Nous introduisons informellement une *syntaxe* pour représenter les spécifications modulaires, et expliquons — également de façon informelle — qu'une spécification modulaire peut être considérée comme la *colimite d'un diagramme*. Nous motivons ensuite l'intérêt des *isomorphismes de construction*. Enfin, nous posons le cadre général de notre travail.

1.1 Préliminaires

Dans cette partie, nous rappelons quelques notions élémentaires de spécification algébrique, en particulier les définitions de signature, morphisme de signatures, algèbre, morphisme d'algèbres, équation et algèbre satisfaisant une équation. Cette présentation, loin d'être exhaustive, est inspirée de [GB90] et de [EM85].

Définition 1.1 (S-ensemble et S-application) Soit un ensemble S.
Un S-ensemble A est une famille indicée par S d'ensembles disjoints $(A_s)_{s\in S}$.
Une S-application f entre deux S-ensembles A et B, notée $f : A \to B$, est une famille indicée par S d'applications $(f_s : A_s \to B_s)_{s\in S}$.

Définition 1.2 (Ensemble des chaînes finies) Soit un ensemble A. L'ensemble des *chaînes finies* sur A est l'ensemble

$$A^* = \bigcup_{n=0}^{\infty} A^n.$$

Notations :

- La chaîne vide est notée ε.
- La chaîne composée des éléments $s_1, s_2, \ldots, s_n$ est notée $s_1 s_2 \ldots s_n$.

L'ensemble des chaînes finies non vides sur A est

$$A^+ = A^* \times A = \bigcup_{n=1}^{\infty} A^n.$$

Les opérations $_^+$ et $_^*$ s'étendent aux applications entre deux ensembles. Soit une application $f : A \to B$. On définit les applications $f^* : A^* \to B^*$ et $f^+ : A^+ \to B^+$ de la façon suivante.

$f^*(\varepsilon) = \varepsilon$;
$f^*(a_1 \ldots a_n) = f(a_1) \ldots f(a_n)$ (pour $n \geq 1$) ;
$f^+(a_1 \ldots a_n) = f(a_1) \ldots f(a_n)$ (pour $n \geq 1$).

On vérifie facilement que pour tout ensemble A, on a

$$(id_A)^* = id_{A^*}$$
$$(id_A)^+ = id_{A^+}$$

et pour toutes applications $f : A \to B$ et $g : B \to C$, on a

$$(g \circ f)^* = g^* \circ f^*$$
$$(g \circ f)^+ = g^+ \circ f^+.$$

Ces deux propriétés signifient que $_^*$ et $_^+$ sont des *foncteurs* $_^*$: **Set** $\to$ **Set** et $_^+$: **Set** $\to$ **Set** dans la catégorie des ensembles **Set**.

Une signature comprend les symboles utilisés pour former des expressions. Plus précisément, une signature est composée d'un ensemble de *symboles de sortes*, qui représentent des types de données, et d'un ensemble de *symboles d'opérateurs*, qui représentent des opérations sur ces données. À chaque symbole d'opérateur est associé un *profil*, qui représente le type des données d'entrée et de sortie.

Définition 1.3 (Signature multi-sortes) Une *signature multi-sortes*, ou plus simplement *signature*, est un triplet (S, Ω, ϖ) où

- S est un ensemble fini, dont les éléments sont appelés *sortes* ;
- Ω est un ensemble fini, dont les éléments sont appelés *opérateurs* ;
- ϖ est une application $\varpi : \Omega \to S^+$, qui à chaque opérateur ω de Ω associe une chaîne non vide sur S, appelée *profil* de ω.

Notation : si $\varpi(\omega) = s_1 \ldots s_n s$ $(n \geq 0)$, on note $\omega : s_1, \ldots s_n \to s$. Si $n = 0$, on dit que ω est une *constante*, et on note $\omega : \to s$.

Remarque 1.1 Une signature est souvent définie comme un ensemble de sortes S accompagné d'une famille indicée par $S^* \times S$ d'ensembles d'opérateurs. Nous avons préféré considérer ici un unique ensemble aplati Ω d'opérateurs, le profil étant décrit par l'application $\varpi : \Omega \to S^+$. Nous avons choisi cette définition parce qu'elle permet de définir facilement la somme amalgamée de deux signatures à partir de la définition de la somme amalgamée de deux ensembles (cf. section 1.4).

Exemple 1.1 (Exemples de signatures)

- La signature la plus simple est la signature vide, notée $\Sigma_{\varnothing}$, telle que l'ensemble des sortes et l'ensemble des opérateurs sont vides.

- Un exemple classique de signature est la signature $\Sigma_{Nat} = (S_{Nat}, \Omega_{Nat}, \varpi_{Nat})$ des entiers naturels où S_{Nat} et Ω_{Nat} et ϖ_{Nat} sont définis de la façon suivante :

  $S_{Nat} = \{nat\}$;
  $\Omega_{Nat} = \{0, succ, +\}$;
  $\varpi_{Nat}(0) = nat$;
  $\varpi_{Nat}(succ) = nat\ nat$;
  $\varpi_{Nat}(+) = nat\ nat\ nat$.

  Dans le langage de spécification LPG, les noms de sortes sont introduits par le mot clé `sorts` et les opérateurs par le mot clé `operators`. La signature Σ_{Nat} est donc définie en LPG par :

```
sorts      nat
operators  0 : ->nat
           succ : nat ->nat
           + : nat, nat ->nat
```

- Voici un autre exemple de signature, la signature Σ_M des monoïdes. Dans cette signature, on a une seule sorte s, une constante 1 et un opérateur binaire $*$. En LPG, on peut définir la signature Σ_M de la façon suivante :

```
sorts      s
operators  1 : ->s
           * : s, s ->s
```

Un *morphisme de signatures* entre deux signatures Σ et Σ' associe à toute sorte de Σ une sorte de Σ' et à tout opérateur de Σ un opérateur de Σ', en conservant les profils des opérateurs.

Définition 1.4 (Morphisme de signatures) Soit deux signatures $\Sigma = (S, \Omega, \varpi)$ et $\Sigma' = (S', \Omega', \varpi')$. Un morphisme de signatures m de Σ vers Σ', noté $m : \Sigma \to \Sigma'$ est un couple (m^S, m^Ω) où

- m^S est une application $m^S : S \to S'$;
- m^Ω est une application $m^\Omega : \Omega \to \Omega'$;

tel que $\varpi' \circ m^\Omega = m^{S+} \circ \varpi$.

Détaillons un peu la condition $\varpi' \circ m^\Omega = m^{S+} \circ \varpi$. Considérons un opérateur $\omega : s_1, \ldots, s_n \to s$ de Ω.

$$\begin{aligned} \varpi'(m^{\Omega}(\omega)) &= m^{S+}(\varpi(\omega)) \\ &= m^{S+}(s_1 \ldots s_n s) \\ &= m^S(s_1) \ldots m^S(s_n) m^S(s) \end{aligned}$$

L'opérateur $m^{\Omega}(\omega)$ de Ω' a donc pour profil $m^{\Omega}(\omega) : m^S(s_1), \ldots, m^S(s_n) \to m^S(s)$.

On peut composer les morphismes de signatures. Soit trois signatures

$$\begin{aligned} \Sigma &= (S, \Omega, \varpi) \\ \Sigma' &= (S', \Omega', \varpi') \\ \Sigma'' &= (S'', \Omega'', \varpi'') \end{aligned}$$

et deux morphismes de signatures

$$\begin{aligned} m &= (m^S, m^{\Omega}) : \Sigma \to \Sigma' \\ n &= (n^S, n^{\Omega}) : \Sigma' \to \Sigma''. \end{aligned}$$

Pour composer m et n, il suffit de composer les applications sur les sortes et les applications sur les opérateurs. Autrement dit,

$$n \circ m = (n^S \circ m^S, n^{\Omega} \circ m^{\Omega}).$$

Cela définit bien un morphisme de signatures car

$$\begin{aligned} & \varpi'' \circ n^{\Omega} \circ m^{\Omega} \\ &= n^{S+} \circ \varpi' \circ m^{\Omega} && (n : \Sigma' \to \Sigma'' \text{ morphisme de signatures}) \\ &= n^{S+} \circ m^{S+} \circ \varpi && (m : \Sigma \to \Sigma' \text{ morphisme de signatures}) \\ &= (n^S \circ m^S)^+ \circ \varpi && (_^+ \text{ est un foncteur}) \end{aligned}$$

On a une catégorie des signatures, notée **Sig**, dont les objets sont les signatures, et les flèches sont les morphismes de signatures.

Exemple 1.2 (Exemples de morphismes de signatures)

- Pour toute signature $\Sigma = (S, \Omega, \varpi)$, il existe un morphisme de signatures de la signature vide $\Sigma_{\varnothing}$ vers Σ. Ce morphisme, noté $j_{\Sigma} = (j_{\Sigma}^S, j_{\Sigma}^{\Omega})$, est constitué de l'application vide de l'ensemble vide des sortes vers S, et de l'application vide de l'ensemble vide des opérateurs vers Ω. De plus, ce morphisme est unique. Dans la catégorie des signatures **Sig**, $\Sigma_{\varnothing}$ est un *objet initial* puisqu'il existe une unique flèche de $\Sigma_{\varnothing}$ vers tout autre objet de **Sig**.

- On a un morphisme de signatures de la signature des monoïdes vers la signature des entiers naturels $m : \Sigma_M \to \Sigma_{Nat}$, qui associe la sorte s de Σ_M à la sorte nat de Σ_{Nat}, la constante 1 à la constante 0, et l'opérateur binaire $*$ à l'opérateur binaire $+$.

  $m^S(s) = nat$;
  $m^{\Omega}(1) = 0$;
  $m^{\Omega}(*) = +$.

Soit une signature Σ. Définir une *algèbre sur* Σ, ou Σ-*algèbre*, consiste à interpréter les sortes de Σ par des ensembles, et les opérateurs de Σ par des applications entre ces ensembles. Une Σ-algèbre est également appelée *modèle* ou *interprétation* de la signature Σ.

Définition 1.5 (Σ-algèbre) Soit une signature $\Sigma = (S, \Omega, \varpi)$. Une Σ-*algèbre* A est définie par :

- un S-ensemble A ;
- pour tout opérateur $\omega : s_1, s_2, \ldots, s_n \to s \in \Omega$, une *application*

$$\omega^A : A_{s_1} \times A_{s_2} \times \cdots \times A_{s_n} \to A_s.$$

Si ω est une constante $\omega : \to s$, ω^A est un élément de A_s.

Définition 1.6 (Morphisme d'algèbres) Soit une signature $\Sigma = (S, \Omega, \varpi)$ et deux algèbres A et B. Un *morphisme d'algèbres* h de A vers B, noté $h : A \to B$ est une S-application $(h_s : A_s \to B_s)_{s \in S}$ telle que pour tout opérateur $\omega : s_1, s_2, \ldots, s_n \to s$ de Ω, pour tout n-uplet $(a_1, a_2, \ldots, a_n)$ de $A_{s_1} \times A_{s_2} \times \cdots \times A_{s_n}$,

$$h_s(\omega^A(a_1, a_2, \ldots a_n)) = \omega^B(h_{s_1}(a_1), h_{s_2}(a_2), \ldots, h_{s_n}(a_n)).$$

Étant donné une signature Σ, on a une catégorie de Σ-algèbres, notée $\mathrm{Alg}(\Sigma)$, dont les objets sont les Σ-algèbres et les flèches sont les morphismes de Σ-algèbres.

Une algèbre importante est l'algèbre des termes, qui permet de former des expressions à partir des symboles de la signature. Nous devons d'abord définir la notion de *variable* sur une signature.

Définition 1.7 (Ensemble de variables sur une signature)
Soit une signature $\Sigma = (S, \Omega, \varpi)$. Un ensemble X de *variables sur la signature* Σ est un S-ensemble fini, tel que pour tout élément $s \in S$, X_s est distinct de l'ensemble des opérateurs Ω.

Définition 1.8 (Algèbre des termes) Soit une signature $\Sigma = (S, \Omega, \varpi)$ et un ensemble de variables X sur Σ. L'algèbre des termes $T_\Sigma(X)$ est définie de la façon suivante.

- Pour toute sorte $s \in S$, $T_\Sigma(X)_s$ est le plus petit ensemble tel que :
  - $x \in X_s \Rightarrow x \in T_\Sigma(X)_s$;
  - $\omega : s_1, s_2, \ldots, s_n \to s \in \Omega,\ t_i \in T_\Sigma(X)_{s_i}\ (i = 1, \ldots, n)$
    $\Rightarrow \omega(t_1, t_2, \ldots, t_n) \in T_\Sigma(X)_s$.

  Les éléments de $T_\Sigma(X)_s$ sont appelés *termes de sorte* s.

- L'interprétation d'un opérateur $\omega : s_1, s_2, \ldots, s_n \to s \in \Omega$ est l'application

$$\begin{array}{rccc} \omega^{T_\Sigma(X)} : & T_\Sigma(X)_{s_1} \times \ldots \times T_\Sigma(X)_{s_n} & \to & T_\Sigma(X)_s \\ & (t_1, \ldots, t_n) & \mapsto & \omega(t_1, \ldots, t_n). \end{array}$$

Lorsque l'ensemble des variables X est vide, on obtient l'algèbre des termes fermés $T_\Sigma = T_\Sigma(\varnothing)$.

Une *affectation* d'un ensemble de variables X dans une Σ-algèbre A associe à toute variable de X_s un élément de A_s. Autrement dit, une affectation *aff* de X dans A est une S-application $aff : X \to A$.

L'algèbre des termes a une propriété importante: c'est une algèbre *libre* dans la classe des Σ-algèbres. Cela signifie que toute affectation de X dans une algèbre A s'étend de façon unique en un morphisme d'algèbres de $T_\Sigma(X)$ vers A.

Théorème 1.1 *(L'algèbre $T_\Sigma(X)$ est libre dans* Alg(Σ)*)*
Notons $i : X \to T_\Sigma(X)$ l'inclusion de X dans $T_\Sigma(X)$. Soit une Σ-algèbre A. Pour toute affectation $aff : X \to A$, il existe un unique morphisme d'algèbres $\overline{aff} : T_\Sigma(X) \to A$ tel que $\overline{aff} \circ i = aff$.

Une équation est tout simplement composée d'un ensemble de variables et de deux termes.

Définition 1.9 (Σ-équation) Soit une signature $\Sigma = (S, \Omega, \varpi)$ et un ensemble de variables X sur Σ. Une *Σ-équation sur Σ* de sorte s est un triplet (X, t_1, t_2) où t_1 et t_2 sont des éléments de $T_\Sigma(X)_s$.

En LPG, une signature peut être accompagnée d'un ensemble d'équations. Toutes ces équations ont le même ensemble de variables, qui sont introduites une seule fois par le mot clé `variables`. Ensuite, les Σ-équations (X, t_1, t_2) sont introduites par le mot clé `equations` et notées $t_1 == t_2$.

Définition 1.10 (Algèbre satisfaisant une équation, Σ-E-algèbre)
Soit une signature $\Sigma = (S, \Omega, \varpi)$. On dit qu'une Σ-algèbre A *satisfait une équation* (X, t_1, t_2) si et seulement si pour toute affectation $aff : X \to A$, $\overline{aff}(t_1) = \overline{aff}(t_2)$.
Étant donné un ensemble d'équations E sur Σ, une *Σ-E-algèbre* est une Σ-algèbre qui satisfait toutes les équations de E.

Définition 1.11 (Fermeture sémantique d'un ensemble d'équations)
Soit une signature Σ et un ensemble d'équations E. La *fermeture sémantique* $\overline{E}$ de E est l'ensemble des équations sur Σ qui sont satisfaites par toute Σ-E-algèbre.

Définition 1.12 (Extension aux termes d'un morphisme de signatures)
Soit deux signatures $\Sigma = (S, \Omega, \varpi)$ et $\Sigma' = (S', \Omega', \varpi')$, ainsi qu'un morphisme de signatures $m : \Sigma \to \Sigma'$. Soit un ensemble de variables X sur Σ, dont les éléments sont distincts des constantes de Ω'.
La *traduction* de X par le morphisme de signatures $m : \Sigma \to \Sigma'$ est le S'-ensemble $X^\#$ tel que pour toute sorte s' de S',

$$X^\#_{s'} = \bigcup_{s' = m^S(s)} X_s.$$

L'*extension* du morphisme de signatures $m : \Sigma \to \Sigma'$ est une famille indicée par S d'applications $m^{\#} = (m^{\#}_s : T_\Sigma(X)_s \to T_{\Sigma'}(X^{\#})_{m^S(s)})_{s \in S}$. Pour toute sorte $s \in S$, l'application $m^{\#}_s : T_\Sigma(X)_s \to T_{\Sigma'}(X^{\#})_{m^S(s)}$ est définie inductivement sur la structure des éléments de $T_\Sigma(X)_s$ de la façon suivante.

- $\forall x \in X_s,\ m^{\#}_s(x) = x\ (x \in X^{\#}_{m^S(s)})$;
- $\forall \omega : s_1, s_2, \ldots s_n \to s,\ \forall t_i \in T_\Sigma(X)_{s_i}\ (i = 1, \ldots, n)$,
$$m^{\#}_s(\omega(t_1, t_2, \ldots, t_n)) = m^{\Omega}(\omega)(m^{\#}_{s_1}(t_1), m^{\#}_{s_2}(t_2), \ldots, m^{\#}_{s_n}(t_n)).$$

Étant donné un terme $t \in T_\Sigma(X)_s$, $m^{\#}(t) = m^{\#}_s(t)$ est appelé *traduction de t par m*.

Exemple 1.3 Considérons le morphisme de signatures $m : \Sigma_M \to \Sigma_{Nat}$ et le terme $1 * x$ de $T_{\Sigma_M}(X)$. La traduction du terme $1 * x$ sur Σ_M par m est le terme sur Σ_{Nat}

$$m^{\#}(1 * x) = m^{\#}(1) + m^{\#}(x) = 0 + x.$$

Définition 1.13 (Traduction d'une équation par un morphisme de signatures) Soit deux signatures $\Sigma = (S, \Omega, \varpi)$ et $\Sigma' = (S', \Omega', \varpi')$, ainsi qu'un morphisme de signatures $m : \Sigma \to \Sigma'$. Soit une équation (X, t_1, t_2) sur Σ. La *traduction* de l'équation (X, t_1, t_2) par le morphisme de signatures m est l'équation $(X^{\#}, m^{\#}(t_1), m^{\#}(t_2))$ sur la signature Σ'.

Exemple 1.4 Considérons par exemple l'équation $(\{x\}, 1 * x, x)$ sur la signature Σ_M. La traduction de cette équation par le morphisme de signatures $m : \Sigma_M \to \Sigma_{Nat}$ est l'équation $(\{x\}, 0 + x, x)$ sur Σ_{Nat}.

1.2 Spécifications

Dans ce paragraphe, nous présentons les spécifications *algébriques*, qui consistent en une *signature* et un ensemble de *formules*. Dans l'exemple des anneaux développé dans ce chapitre, les formules que nous considérons sont uniquement des *équations*. Pour cette raison, nous nous contentons de définir des spécifications équationnelles. Nous verrons en section 1.7 que cette hypothèse peut être affaiblie en se plaçant dans le cadre général des institutions.

Définition 1.14 (Spécification équationnelle) Une *spécification équationnelle* est un couple (Σ, E), où Σ est une signature et E est un ensemble de Σ-équations.

Définition 1.15 (Morphisme de spécifications) Un *morphisme de spécifications* entre deux spécifications (Σ, E) et (Σ', E') est un morphisme de signatures $m : \Sigma \to \Sigma'$ tel que pour toute équation $e \in E$, $m^{\#}(e) \in \overline{E'}$. Rappelons que

- $m^{\#}(e)$ est la *traduction* de l'équation e par le morphisme de signatures m ;
- $\overline{E'}$ est la fermeture sémantique de E', c'est-à-dire l'ensemble des équations sur Σ' qui sont satisfaites dans toute Σ'-E'-algèbre.

Une autre façon de définir un morphisme de spécifications est d'imposer que la traduction par $m^{\#}$ de toute équation de E *appartienne à la théorie équationnelle de* E', c'est-à-dire peut être déduite des équations de E' par déduction en logique équationnelle. Il s'agit d'une définition plus *syntaxique* de la notion de spécification. Nous avons choisi de suivre la définition donnée pour les institutions par J. Goguen et R. Burstall dans [GB90]. Dans le cadre des spécifications algébriques équationnelles, les deux définitions sont équivalentes puisque le système de déduction équationnel est complet. En effet, une équation e est satisfaite dans toute Σ-E-algèbre si et seulement si on peut déduire e des équations de E par raisonnement équationnel.

On a une catégorie des spécifications équationnelles, notée **Spec**, dont les objets sont les spécifications (équationnelles) et les flèches sont les morphismes de spécifications.

Exemple 1.5

- La spécification la plus simple est la spécification vide $(\Sigma_\varnothing, \varnothing)$, construite sur la signature vide $\Sigma_\varnothing$, et dont l'ensemble des équations est vide. Cette spécification est *initiale* dans la catégorie des spécifications, car pour toute spécification Sp, il existe un unique morphisme de spécifications de la spécification vide vers Sp.

- Voici un exemple de spécification des entiers écrit en LPG :

```
type        Nat
sorts       nat
operators   0 : ->nat
            succ : nat ->nat
            + : nat, nat ->nat
variables   x, y : nat
equations   0 + y == y
            s(x) + y == s(x + y)
```

Le langage LPG permet de spécifier à la fois des *types*, comme le type `Nat`, qui ont pour interprétation une algèbre initiale (ou pour les types génériques une algèbre libre), et des *propriétés*, qui ont pour interprétation une classe d'algèbres satisfaisant les équations de la spécification. On peut consulter [Rey87] pour avoir une description plus détaillée de la sémantique de LPG. Dans l'exemple développé ici, nous utiliserons uniquement les propriétés, sans importation de type, ce qui signifie qu'un *modèle* ou une *interprétation* d'une spécification est tout simplement une algèbre qui satisfait les équations de la spécification. Le langage LPG permet également de déclarer des morphismes de spécifications. Un morphisme d'une spécification Sp vers une spécification Sp' est défini au moment de la déclaration du codomaine Sp' par une instruction commençant par le mot clé `satisfies` ou par le mot clé `inherits`. Le langage LPG permet également de considérer les compositions des morphismes de spécifications déclarés. Les règles de composition de ces morphismes et leur sémantique associée ont été décrits dans [BO95].

Nous allons maintenant présenter les spécifications équationnelles et les morphismes de spécifications qui nous permettront de définir plusieurs spécifications modulaires des anneaux. Rappelons qu'un ensemble A, muni de deux opérations binaires $+$ et $*$ est un *anneau* si et seulement si A muni de l'opération $+$ est un groupe commutatif, A muni de l'opération $*$ est un monoïde, et l'opération $*$ est distributive par rapport à l'opération $+$.

Nous commençons par définir deux spécifications S et B qui comportent uniquement une signature et aucune équation. La spécification S contient une sorte unique `s`. Un modèle de S est donc un ensemble quelconque.

$S =$

```
property  UNE-SORTE
sorts     s
```

La spécification B contient une sorte `s` et un opérateur binaire `op`. Un modèle de B consiste donc en un ensemble muni d'une opération binaire. Nous avons également spécifié un morphisme de spécifications $s : S \to B$ par la proposition `satisfies UNE-SORTE[s]` dans B. Le morphisme de spécifications s associe la sorte `s` de S à la sorte `s` de B.

$B =$

```
property    OP-BIN
sorts       s
operators   op : s, s ->s
satisfies   UNE-SORTE[s]
```

Nous définissons maintenant une spécification M pour les monoïdes. La signature de la spécification M comporte une sorte `s`, une opération binaire `*`, et une constante `1`. La spécification M comporte également trois équations qui indiquent que `1` est élément neutre pour `*`, et que `*` est associative. Un modèle de M est donc un ensemble muni d'une opération binaire associative et d'un élément neutre. Il s'agit donc bien d'un monoïde.

Nous avons également un morphisme de spécifications $b : B \to M$. Le morphisme b associe à la sorte `s` de B la sorte `s` de M et à l'opérateur `op` de B l'opérateur `*` de M. Ce morphisme de signatures est bien un morphisme de spécifications, puisque la spécification de l'opérateur binaire B ne comporte aucune équation.

$M =$

```
property    MONOIDE
sorts       s
operators   * : s, s ->s
            1 : ->s
variables   x, y, z : s
equations   1 * x == x
            x * 1 == x
            (x * y) * z == x * (y * z)
satisfies   OP-BIN[s,*]
```

Pour obtenir un groupe commutatif, nous devons ajouter une opération inverse `i`, et la commutativité de l'opérateur binaire. Cela nous donne la spécification G.

$G =$

```
property    GROUPE-COMM
sorts       s
operators   + : s, s ->s
            0 : ->s
            i : s ->s
variables   x, y : s
equations   x + y == y + x
            i(x) + x == 0
inherits    MONOIDE[s,+,0]
```

Nous spécifions également un morphisme de signatures de la signature de M vers la signature de G, $m : \Sigma_M \to \Sigma_G$, par la proposition `inherits MONOIDE[s,+,0]`. Ce morphisme de signatures associe la sorte `s` de M à la sorte s de G, l'opérateur `*` de M à l'opérateur `+` de G, et la constante `1` de M à la constante `0` de G. Le mot clé `inherits` signifie que l'on "recopie" les équations de la spécification M dans la spécification G via le morphisme de signatures $m^{\#}$. Autrement dit, G contient implicitement les équations

```
variables   x, y, z : s
equations   0 + x == x
            x + 0 == x
            (x + y) + z == x + (y + z)
```

Par conséquent, le morphisme de signatures $m : \Sigma_M \to \Sigma_G$ est un morphisme de spécifications $m : M \to G$.

Enfin, la propriété D spécifie deux opérateurs `+` et `*` tels que `*` est distributif par rapport à `+`, ainsi que deux morphismes de spécifications $m_+, m_* : B \to D$.

$D =$

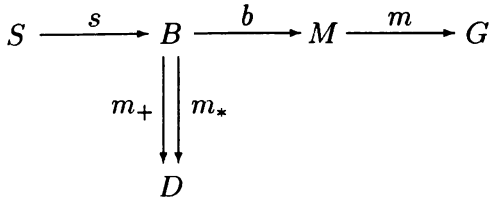
```
property    DISTRIBUTIVITE
sorts       s
operators   +, * : s, s ->s
variables   x, y, z : s
equations   x * (y + z) == (x * y) + (x * z)
satisfies   OP-BIN[s,+], OP-BIN[s,*]
```

Les deux morphismes $m_+, m_* : B \to D$ sont définis par :

$$
\begin{aligned}
m_+(\mathtt{s}) &= \mathtt{s};\\
m_+(\mathtt{op}) &= \mathtt{+};\\
m_*(\mathtt{s}) &= \mathtt{s};\\
m_*(\mathtt{op}) &= \mathtt{*}.
\end{aligned}
$$

Finalement, pour résumer, nous avons déclaré cinq spécifications S, B, M, G, D, et cinq morphismes de spécifications s, b, m, m_+, m_*. Nous pouvons évidemment

considérer également les morphismes de spécifications identités id_S, id_B, id_M, id_G et id_D sur ces cinq spécifications, ainsi que toutes les compositions possibles pour les morphismes de spécifications. On vérifie facilement que l'on obtient ainsi quinze morphismes de spécifications différents. Autrement dit, nous considérons une *catégorie de spécifications*, appelée $\mathcal{C}_0$, qui comporte ces cinq spécifications et ces quinze morphismes de spécifications.



Squelette de la catégorie $\mathcal{C}_0$

Nous avons défini les spécifications équationnelles, qui forment une catégorie **Spec**, ainsi qu'une sous-catégorie finie $\mathcal{C}_0$ de **Spec**, que nous appellerons catégorie des spécifications *de base*. La catégorie $\mathcal{C}_0$ représente l'ensemble des spécifications disponibles en bibliothèque. Nous nous intéressons ensuite uniquement aux spécifications que l'on peut construire en assemblant des spécifications de base.

1.3 Composition de spécifications en LPG

Une spécification *modulaire* est soit une spécification de base, soit une spécification obtenue par un assemblage de plusieurs spécifications modulaires. Nous parlerons de *composition* de spécifications pour décrire l'assemblage de plusieurs spécifications permettant de construire une spécification plus complexe.

La composition de spécifications en LPG est réalisée par ce qu'on appelle une *combinaison* de spécifications. Pour réaliser cette opération, il faut définir une signature — la signature de la spécification résultat — et des morphismes de spécifications des spécifications combinées vers la spécification résultat.

Nous pouvons par exemple écrire une spécification des anneaux en combinant les propriétés `MONOIDE`, `DISTRIBUTIVITE` et `GROUP-COMM` de la façon suivante.

$A_1 =$

```
property    ANNEAU
sorts       s
operators   +, * : s, s ->s
            0, 1 : ->s
            i : s ->s
combines    MONOIDE[s,*,1],
            DISTRIBUTIVITE[s,+,*],
            GROUPE-COMM[s,+,0,i]
```

Sémantiquement, cette combinaison signifie que chaque fois qu'on a un modèle de MONOIDE, un modèle de DISTRIBUTIVITE et un modèle de GROUPE-COMM, tels que *ces trois modèles ont la même interprétation pour la sorte* s, *que l'opérateur du monoïde est le second opérateur de la distributivité et l'opérateur du groupe commutatif est le premier opérateur de la distributivité*, alors on a un modèle de la spécification ANNEAU. Ces *partages* de sorte et d'opérateurs sont évidemment cruciaux. Il ne suffit pas d'avoir un monoïde, un groupe commutatif et deux opérateurs distributifs l'un par rapport à l'autre pour avoir un anneau. Par exemple, si nous avions écrit

```
combines ... DISTRIBUTIVITE[s,*,+]
```

à la place de

```
combines ... DISTRIBUTIVITE[s,+,*]
```

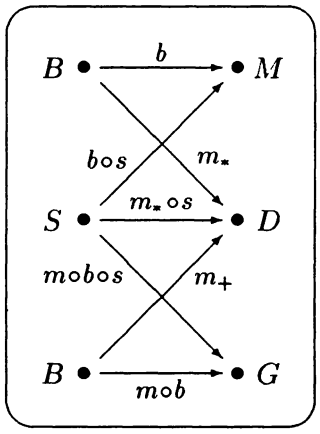
nous n'aurions pas spécifié un anneau. (La multiplication est distributive par rapport à l'addition, et non l'inverse.)

En LPG, les partages entre spécifications, en particulier lors de la construction de combinaisons de propriétés, sont exprimés de façon "implicite", par des morphismes de spécifications. En pratique, on voit que la sorte s est partagée par les trois spécifications par l'apparition de s dans la définition des trois morphismes.

```
combines  MONOIDE[s,*,1],
          DISTRIBUTIVITE[s,+,*],
          GROUPE-COMM[s,+,0,i]
```

Dans notre travail, ces partages effectués lors de la composition des spécifications sont exprimés par des *colimites* de *diagrammes*. La colimite est un concept de théorie des catégories qui sera présenté dans le chapitre 2. Cette construction permet, en particulier dans le cadre des spécifications algébriques, de modéliser l'assemblage de plusieurs objets en rendant explicites certains partages entre ces objets.

Dans l'exemple ci-dessus, la spécification modulaire A_1 correspond à la colimite du diagramme $\overline{\delta_1}$.

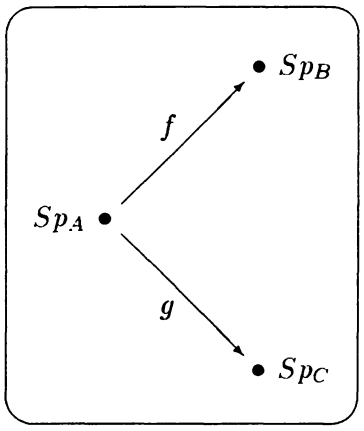


$\overline{\delta_1}$

Voici l'interprétation intuitive de ce diagramme :

- Les trois nœuds étiquetés par les spécifications M, D et G, qui apparaissent comme des puits du graphe (aucun arc n'a pour source un nœud étiqueté par M, D ou G) sont les spécifications que l'on assemble.
- Les trois nœuds étiquetés par B, S et B indiquent les partages.
  - La sorte **s** est partagée par les trois spécifications M, D et G. Ce partage est modélisé par les morphismes $b \circ s : S \to M$, $m_* \circ s : S \to D$ et $m \circ b \circ s : S \to G$.
  - L'opérateur binaire * est partagé par les spécifications M et D. Ce partage est modélisé par les morphismes $b : B \to M$ et $m_* : B \to D$.
  - L'opérateur binaire + est partagé par les spécifications G et D. Ce partage est modélisé par les morphismes $m \circ b : B \to G$ et $m_+ : B \to D$.

Par la suite, nous privilégions une construction particulière de colimite : la somme amalgamée. La somme amalgamée correspond à la colimite d'un diagramme $\overline{\pi}$ qui ne comporte que trois spécifications Sp_A, Sp_B et Sp_C, liées par deux morphismes de spécifications $f : Sp_A \to Sp_B$ et $g : Sp_A \to Sp_C$.



$\overline{\pi}$

La colimite de ce diagramme correspond à la composition des spécifications Sp_B et Sp_C où les partages entre ces deux spécifications sont indiqués par les deux morphismes f et g.

1.4 Sommes amalgamées

Dans cette section, nous définissons la somme amalgamée de deux ensembles, puis de deux signatures, et enfin de deux spécifications. Autrement dit, nous présentons la somme amalgamée dans trois catégories particulières : **Set** (catégorie des ensembles), **Sig** (catégorie des signatures) et **Spec** (catégorie des spécifications). La définition générale sera revue dans le chapitre 2.

1.4.1 Somme amalgamée de deux ensembles

Considérons trois ensembles A, B, et C, ainsi que deux applications $f : A \to B$ et $g : A \to C$. Intuitivement, faire la somme amalgamée de B et C par rapport à f et g consiste à faire l'union disjointe de B et C et à fusionner les éléments qui sont images par f et g d'un même élément a de A.
Plus formellement, on peut définir la somme disjointe de B et C par l'ensemble

$$B + C = (\{1\} \times B) \cup (\{2\} \times C).$$

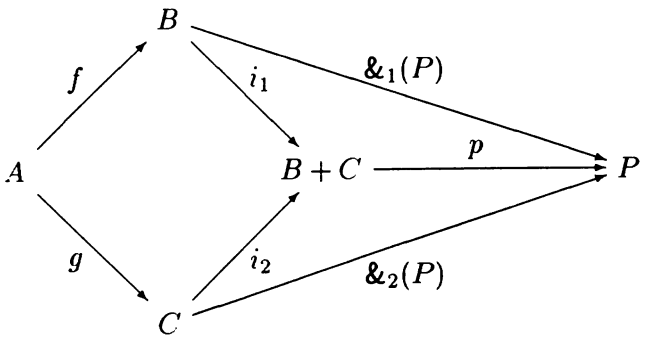
Il y a évidemment un choix de codage dans cette définition. L'idée est de faire la réunion des deux ensembles B et C après avoir renommé les éléments communs aux deux ensembles. Nous avons également deux applications $i_1 : B \to B + C$ et $i_2 : C \to B + C$ définies par

$$\begin{array}{rccc} i_1 : & B & \to & B+C \\ & b & \mapsto & (1,b) \end{array} \qquad \begin{array}{rccc} i_2 : & C & \to & B+C \\ & c & \mapsto & (2,c) \end{array}$$

Puis on définit une relation $\mathcal{R}$ sur l'ensemble $B + C$:

$$\mathcal{R} = \{((1, f(a)), (2, g(a))) \ ; \ \forall a \in A\}.$$

On considère ensuite la relation d'équivalence $\overline{\mathcal{R}}$ engendrée par $\mathcal{R}$ et l'ensemble quotient $P = (B + C)/\overline{\mathcal{R}}$. On a une application de projection $p : B + C \to P$, qui à tout élément de $B + C$ associe sa classe d'équivalence modulo $\overline{\mathcal{R}}$.



Notations

- L'ensemble P est noté **push** (A, B, C, f, g).
- L'application $p \circ i_1 : B \to P$ est notée $\&_1(A, B, C, f, g)$.
- L'application $p \circ i_2 : C \to P$ est notée $\&_2(A, B, C, f, g)$.
- Pour alléger un peu les notations, si $P \equiv$ **push** (A, B, C, f, g), nous écrirons
  - $\&_1(P)$ à la place de $\&_1(A, B, C, f, g)$;

– $\&_2(P)$ à la place de $\&_2(A, B, C, f, g)$.

Définition 1.16 (Somme amalgamée dans **Set**)
La somme amalgamée de B et C par rapport à f et g est le triplet

$$(\mathsf{push}\,(A, B, C, f, g), \&_1(A, B, C, f, g), \&_2(A, B, C, f, g)).$$

Par abus de langage, l'ensemble $\mathsf{push}\,(A, B, C, f, g)$ sera également appelé somme amalgamée. Il ne faut pas oublier qu'en réalité les deux applications $\&_1(A, B, C, f, g)$ et $\&_2(A, B, C, f, g)$ font partie de la définition d'une somme amalgamée.

Nous formulons maintenant la propriété générale des sommes amalgamées en théorie des catégories. Nous reverrons cette propriété dans le chapitre 2.

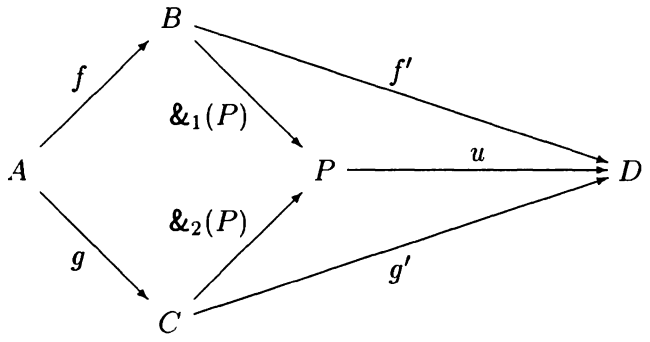
Propriété 1.1 *(Propriété caractéristique des sommes amalgamées)*
Une somme amalgamée

$$(\mathsf{push}\,(A, B, C, f, g), \&_1(A, B, C, f, g), \&_2(A, B, C, f, g))$$

satisfait les propriétés suivantes.

1. $\&_1(A, B, C, f, g) \circ f = \&_2(A, B, C, f, g) \circ g$;
2. *pour tout objet D et flèches $f' : B \to D$, $g' : C \to D$ telles que $f' \circ f = g' \circ g$, il existe une flèche unique u : $\mathsf{push}\,(A, B, C, f, g) \to D$ telle que*

$$\begin{cases} u \circ \&_1(A, B, C, f, g) = f' \\ u \circ \&_2(A, B, C, f, g) = g'. \end{cases}$$



Dans la catégorie des ensembles **Set**, les objets sont les ensembles, et les flèches sont les applications entre deux ensembles. La propriété 1.1 se reformule donc de la façon suivante :

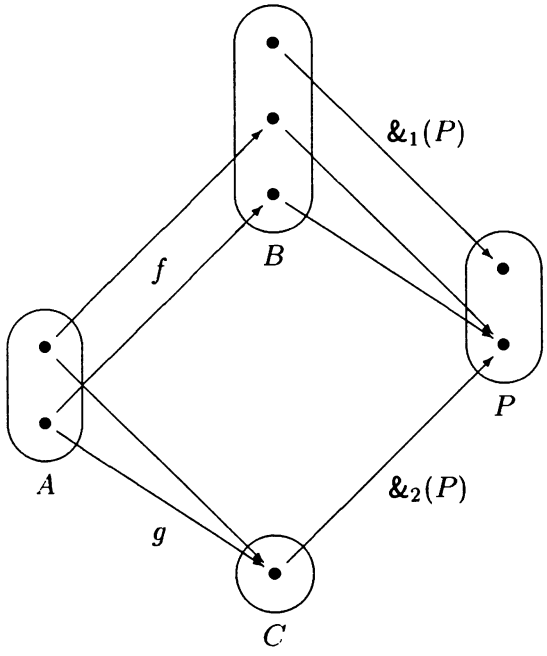
1. $\&_1(A, B, C, f, g) \circ f = \&_2(A, B, C, f, g) \circ g$
   (Il s'agit d'une égalité entre deux applications de A vers $\mathsf{push}\,(A, B, C, f, g)$.)

2. Pour tout *ensemble* D et *applications* $f' : B \to D$, $g' : C \to D$ telles que $f' \circ f = g' \circ g$, il existe une *application* unique u : **push** $(A, B, C, f, g) \to D$ telle que

$$\begin{cases} u \circ \&_1(A, B, C, f, g) = f' \\ u \circ \&_2(A, B, C, f, g) = g'. \end{cases}$$

Exemple 1.6 Soit $A = \{a_1, a_2\}$, $B = \{x, y, z\}$ et $C = \{x\}$.
Soit $f = \{(a_1, x), (a_2, y)\}$ et $g = \{(a_1, x), (a_2, x)\}$.

$B + C = \{(1, x), (1, y), (1, z), (2, x)\}$
$P =$ **push** $(A, B, C, f, g) = \{\{(1, x), (1, y), (2, x)\}, \{(1, z)\}\}$
$\&_1(P)(x) = \&_1(P)(y) = \{(1, x), (1, y), (2, x)\}$
$\&_1(P)(z) = \{(1, z)\}$
$\&_2(P)(x) = \{(1, x), (1, y), (2, x)\}$



Remarque 1.2 Nous avons en réalité défini ici un *codage* de la somme amalgamée pour les ensembles. Dans la théorie des catégories, la somme amalgamée est définie à un isomorphisme près. Dans l'exemple 1.6, en prenant pour P un ensemble quelconque à deux éléments, la propriété 1.1 est encore vérifiée.

1.4.2 Somme amalgamée de deux signatures

Nous définissons la somme amalgamée de deux signatures à partir de la définition de la somme amalgamée de deux ensembles. Soit trois signatures

$$\Sigma_A = (S_A, \Omega_A, \varpi_A)$$
$$\Sigma_B = (S_B, \Omega_B, \varpi_B)$$
$$\Sigma_C = (S_C, \Omega_C, \varpi_C)$$

et deux morphismes de signatures

$$f = (f^S, f^\Omega) : \Sigma_A \to \Sigma_B$$
$$g = (g^S, g^\Omega) : \Sigma_A \to \Sigma_C.$$

On définit la signature $\Sigma_P = (S_P, \Omega_P, \varpi_P)$, somme amalgamée des signatures Σ_B et Σ_C par rapport aux morphismes $f : \Sigma_A \to \Sigma_B$ et $g : \Sigma_A \to \Sigma_C$ de la façon suivante. L'ensemble S_P est la somme amalgamée des ensembles S_B et S_C par rapport à f^S et g^S. Autrement dit,

$$S_P = \mathsf{push}\,(S_A, S_B, S_C, f^S, g^S).$$

L'ensemble Ω_P est la somme amalgamée des ensembles Ω_B et Ω_C par rapport à f^Ω et g^Ω. Autrement dit,

$$\Omega_P = \mathsf{push}\,(\Omega_A, \Omega_B, \Omega_C, f^\Omega, g^\Omega).$$

Il reste à définir $\varpi_P : \Omega_P \to S_P^+$ pour obtenir la signature Σ_P. On a l'égalité

$$\&_1(S_P)^+ \circ \varpi_B \circ f^\Omega = \&_2(S_P)^+ \circ \varpi_C \circ g^\Omega.$$

En effet,

$$\begin{array}{lll}
\&_1(S_P)^+ \circ \varpi_B \circ f^\Omega & & \\
\quad = \&_1(S_P)^+ \circ f^{S+} \circ \varpi_A & (f : \Sigma_A \to \Sigma_B \text{ morphisme de signatures}) \\
\quad = (\&_1(S_P) \circ f^S)^+ \circ \varpi_A & (_^+ \text{ foncteur}) \\
\quad = (\&_2(S_P) \circ g^S)^+ \circ \varpi_A & (\text{propriété des sommes amalgamées}) \\
\quad = \&_2(S_P)^+ \circ g^{S+} \circ \varpi_A & (_^+ \text{ foncteur}) \\
\quad = \&_2(S_P)^+ \circ \varpi_C \circ g^\Omega & (g : \Sigma_A \to \Sigma_C \text{ morphisme de signatures})
\end{array}$$

donc, comme $\Omega_P = \mathsf{push}\,(\Omega_A, \Omega_B, \Omega_C, f^\Omega, g^\Omega)$ et d'après la propriété caractéristique des sommes amalgamées, il existe une unique application $\varpi_P : \Omega_P \to S_P^+$ telle que

$$\varpi_P \circ \&_1(\Omega_P) = \&_1(S_P)^+ \circ \varpi_B \quad (1)$$
$$\varpi_P \circ \&_2(\Omega_P) = \&_2(S_P)^+ \circ \varpi_C \quad (2)$$

On pose

$$\mathsf{push}\,(\Sigma_A, \Sigma_B, \Sigma_C, f, g) = \Sigma_P\,;$$
$$\&_1(\Sigma_P) = \&_1(\Sigma_A, \Sigma_B, \Sigma_C, f, g) = (\&_1(S_P), \&_1(\Omega_P))\,;$$
$$\&_2(\Sigma_P) = \&_2(\Sigma_A, \Sigma_B, \Sigma_C, f, g) = (\&_2(S_P), \&_2(\Omega_P)).$$

D'après les égalités (1) et (2) ci-dessus, $\&_1(\Sigma_P)$ et $\&_2(\Sigma_P)$ sont bien des morphismes de signatures.

Théorème 1.2 *Le triplet*

$$(\mathsf{push}\,(\Sigma_A, \Sigma_B, \Sigma_C, f, g), \&_1(\Sigma_A, \Sigma_B, \Sigma_C, f, g), \&_2(\Sigma_A, \Sigma_B, \Sigma_C, f, g))$$

est une somme amalgamée dans la catégorie des signatures **Sig**.

Preuve. On vérifie facilement que la propriété 1.1 est satisfaite. □

Exemple 1.7 Considérons les signatures Σ_S, Σ_M et Σ_G correspondant respectivement aux spécifications S, M et G (propriétés **UNE_SORTE**, **MONOIDE** et **GROUPE_COMM**). Nous avons également deux morphismes de signatures $b \circ s : \Sigma_S \to \Sigma_M$ et $m \circ b \circ s : \Sigma_S \to \Sigma_G$.
Une somme amalgamée de Σ_M et Σ_G par rapport à $b \circ s$ et $m \circ b \circ s$ est la signature Σ_P suivante :

```
sorts       s
operators   +, * : s, s ->s
            0, 1 : ->s
            i : s ->s
```

On doit fusionner la sorte `s` de Σ_M avec la sorte `s` de Σ_G. Remarquons que nous n'avons pas été obligés de renommer certaines sortes ou opérateurs. En effet, les deux sortes à fusionner ont le même nom dans les deux signatures Σ_M et Σ_G, et les opérateurs que l'on doit distinguer ont des noms distincts dans les deux signatures. Finalement comme la signature partagée Σ_S est l'*intersection* des signatures Σ_M et Σ_G, et comme les deux morphismes de signatures sont les inclusions correspondantes, la somme amalgamée Σ_P est la *réunion* de Σ_M et Σ_G.

1.4.3 Somme amalgamée de deux spécifications

On définit la somme amalgamée de deux spécifications de façon similaire : la signature de la spécification résultat est la somme amalgamée des signatures, et l'ensemble des équations de la spécification résultat est "l'union" (après traduction) des équations des deux spécifications.

Soit trois spécifications

$$Sp_A = (\Sigma_A, E_A)$$
$$Sp_B = (\Sigma_B, E_B)$$
$$Sp_C = (\Sigma_C, E_C)$$

et deux morphismes de spécifications

$$f : Sp_A \to Sp_B$$
$$g : Sp_A \to Sp_C.$$

On définit la spécification $Sp_P = (\Sigma_P, E_P)$ somme amalgamée des spécifications Sp_B et Sp_C par rapport aux morphismes de spécifications f et g de la façon suivante.
La signature Σ_P est la somme amalgamée des signatures :

$$\Sigma_P = \mathsf{push}\,(\Sigma_A, \Sigma_B, \Sigma_C, f, g).$$

On note

$$\&_1(\Sigma_P) = \&_1(\Sigma_A, \Sigma_B, \Sigma_C, f, g)\,;$$
$$\&_2(\Sigma_P) = \&_2(\Sigma_A, \Sigma_B, \Sigma_C, f, g).$$

Intuitivement, E_P contient les équations de E_B et de E_C. Comme les équations de E_B et de E_C sont respectivement des équations sur les signatures Σ_B et Σ_C, on les traduit respectivement par les morphismes de signatures $\&_1(\Sigma_P) : \Sigma_B \to \Sigma_P$ et $\&_2(\Sigma_P) : \Sigma_C \to \Sigma_P$. On pose donc

$$E_P = \{(X^\#, \&_1(\Sigma_P)^\#(t), \&_1(\Sigma_P)^\#(t')) \ ; \ \forall (X,t,t') \in E_B\} \\ \cup \{(X^\#, \&_2(\Sigma_P)^\#(t), \&_2(\Sigma_P)^\#(t')) \ ; \ \forall (X,t,t') \in E_C\}.$$

Remarque 1.3 Les équations présentes dans E_A ne jouent pas directement de rôle. En effet, elles sont déjà "contenues" dans E_B et E_C puisque f et g sont des morphismes *de spécifications.*

Lemme 1.1 *Les morphismes de signatures*

$$\&_1(\Sigma_P) : \Sigma_B \to \Sigma_P$$
$$\&_2(\Sigma_P) : \Sigma_C \to \Sigma_P$$

sont des morphismes de spécifications.

Preuve. C'est évident, par définition de E_P. □

On pose

$$\&_1(Sp_P) = \&_1(Sp_A, Sp_B, Sp_C, f, g) = \&_1(\Sigma_A, \Sigma_B, \Sigma_C, f, g) : Sp_B \to Sp_P ;$$
$$\&_2(Sp_P) = \&_2(Sp_A, Sp_B, Sp_C, f, g) = \&_2(\Sigma_A, \Sigma_B, \Sigma_C, f, g) : Sp_C \to Sp_P.$$

Théorème 1.3 *Le triplet*

$$(\mathsf{push}\,(Sp_A, Sp_B, Sp_C, f, g), \&_1(Sp_A, Sp_B, Sp_C, f, g), \&_2(Sp_A, Sp_B, Sp_C, f, g))$$

est une somme amalgamée dans la catégorie des spécifications **Spec**.

Preuve. On vérifie facilement que la propriété 1.1 est satisfaite. □

Exemple 1.8 Pour poursuivre l'exemple 1.7, on fait la somme amalgamée des spécifications M et G par rapport aux morphismes de spécifications $b \circ s : S \to M$ et $m \circ b \circ s : S \to G$. Cela nous donne la spécification P suivante :

$$P = \mathsf{push}\,(S, M, G, b \circ s, m \circ b \circ s).$$

Détaillons le contenu de la spécification P.

$P =$

```
property    PSEUDO_ANNEAU
sorts       s
operators   +, * : s, s ->s
            0, 1 : ->s
            i : s ->s
variables   x, y, z : s
equations   1 * x == x
            x * 1 == x
            (x * y) * z == x * (y * z)
            0 + x == x
            x + 0 == x
            (x + y) + z == x + (y + z)
            x + y == y + x
            i(x) + x == 0
```

Nous avons appelé cette spécification PSEUDO_ANNEAU car il s'agit d'une spécification des anneaux sans distributivité entre les opérateurs * et +.

1.4.4 Syntaxe abstraite pour les constructions modulaires

Nous récapitulons la syntaxe abstraite qui nous permet de représenter des constructions modulaires. Les deux constructions autorisées sont la spécification vide et la somme amalgamée de deux spécifications. En effet, ces deux seules constructions permettent de simuler la construction de toute colimite finie.

- La spécification vide $(\Sigma_\varnothing, \varnothing)$ est notée $\emptyset$.

  L'unique morphisme de spécifications de $\emptyset$ vers une spécification quelconque Sp est noté $\mathrm{j}_{Sp} : \emptyset \to Sp$.

- Étant donné trois spécifications Sp_A, Sp_B, Sp_C et deux morphismes de spécifications $f : Sp_A \to Sp_B$ et $g : Sp_A \to Sp_C$, la somme amalgamée de Sp_B et Sp_C par rapport à f et g est donnée par une spécification

  $$\mathsf{push}\ (Sp_A, Sp_B, Sp_C, f, g)$$

  et deux morphismes de spécifications

  $$\begin{cases} \&_1(Sp_A, Sp_B, Sp_C, f, g) : Sp_B \to \mathsf{push}\ (Sp_A, Sp_B, Sp_C, f, g) \\ \&_2(Sp_A, Sp_B, Sp_C, f, g) : Sp_C \to \mathsf{push}\ (Sp_A, Sp_B, Sp_C, f, g). \end{cases}$$

  De plus, étant donné une spécification Sp_D et deux flèches $f' : Sp_B \to Sp_D$ et $g' : Sp_C \to Sp_D$ telles que $f' \circ f = g' \circ g$, l'unique morphisme de spécifications

  $$u : \mathsf{push}\ (Sp_A, Sp_B, Sp_C, f, g) \to Sp_D$$

  tel que

  $$\begin{cases} u \circ \&_1(A, B, C, f, g) = f' \\ u \circ \&_2(A, B, C, f, g) = g' \end{cases}$$

  est notée $\mathsf{up}\ (A, B, C, D, f, g, f', g') : \mathsf{push}\ (Sp_A, Sp_B, Sp_C, f, g) \to Sp_D$.

1.4.5 Somme disjointe de deux spécifications

Il est possible de coder la somme disjointe de deux spécifications en utilisant la spécification vide et une somme amalgamée. Soit deux spécifications Sp_A et Sp_B. La somme disjointe de Sp_A et Sp_B est la somme amalgamée de Sp_A et Sp_B par rapport à j_{Sp_A} et j_{Sp_B}.

$$Sp_A + Sp_B = \mathsf{push}\,(\emptyset, Sp_A, Sp_B, \mathrm{j}_{Sp_A}, \mathrm{j}_{Sp_B})$$

De plus, les deux injections de Sp_A vers $Sp_A + Sp_B$ et de Sp_B vers $Sp_A + Sp_B$ sont les deux flèches

$$\&_1(\emptyset, Sp_A, Sp_B, \mathrm{j}_{Sp_A}, \mathrm{j}_{Sp_B}) : Sp_A \to Sp_A + Sp_B\,;$$
$$\&_2(\emptyset, Sp_A, Sp_B, \mathrm{j}_{Sp_A}, \mathrm{j}_{Sp_B}) : Sp_B \to Sp_A + Sp_B.$$

Exemple 1.9 Considérons par exemple la somme disjointe B_2 de deux opérateurs binaires.

$$B_2 = \mathsf{push}\,(\emptyset, B, B, \mathrm{j}_B, \mathrm{j}_B)$$

Cette somme amalgamée correspond à la propriété LPG suivante

$B_2 =$

```
property    DEUX_OP_BIN
sorts       s1, s2
operators   op1 : s1 ->s1
            op2 : s2 ->s2
```

Les deux morphismes de spécifications $\&_1(B_2) : B \to B_2$ et $\&_2(B_2) : B \to B_2$ sont définis par

$$\begin{aligned}
\&_1(B_2)(\mathtt{s}) &= \mathtt{s1}\,;\\
\&_1(B_2)(\mathtt{op}) &= \mathtt{op1}\,;\\
\&_2(B_2)(\mathtt{s}) &= \mathtt{s2}\,;\\
\&_2(B_2)(\mathtt{op}) &= \mathtt{op2}.
\end{aligned}$$

Nous avons deux morphismes de spécifications $m_* : B \to D$ et $m_+ : B \to D$. Ces deux morphismes vérifient l'égalité $m_* \circ \mathrm{j}_B = m_+ \circ \mathrm{j}_B$ car $\emptyset$ est initial dans la catégorie **Spec**. Par conséquent, il existe un unique morphisme de spécifications

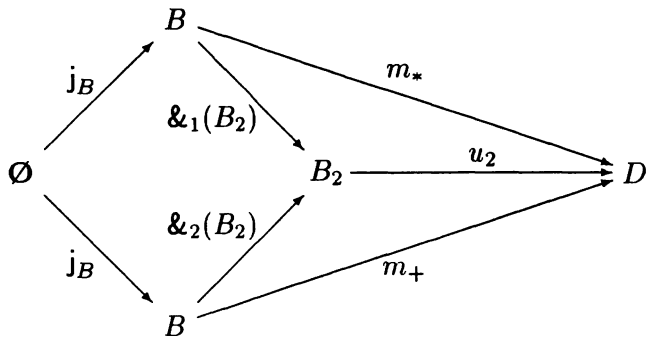
$$u_2 = \mathsf{up}\,(\emptyset, B, B, D, \mathrm{j}_B, \mathrm{j}_B, m_*, m_+) : B_2 \to D$$

tel que

$$\begin{cases} u_2 \circ \&_1(B_2) = m_* \\ u_2 \circ \&_2(B_2) = m_+. \end{cases}$$

Le morphisme de spécifications $u_2 : B_2 \to D$ est défini par

$$\begin{aligned}
u_2(\mathtt{s1}) &= \mathtt{s}\,;\\
u_2(\mathtt{s2}) &= \mathtt{s}\,;\\
u_2(\mathtt{op1}) &= \mathtt{*}\,;\\
u_2(\mathtt{op2}) &= \mathtt{+}.
\end{aligned}$$



1.5 Spécifications modulaires d'anneaux

Nous avons vu en section 1.3 un exemple de spécification modulaire des anneaux, en combinant les trois spécifications M, G et D. On peut coder cette combinaison en utilisant deux sommes amalgamées successives. Par exemple, on peut commencer par combiner M et D, puis combiner le résultat avec G.

$MD =$

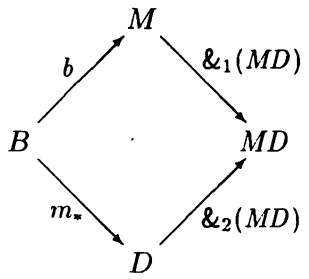
```
property    MONOIDE-DISTRIBUTIF
sorts       s
operators   +, * : s, s ->s
            1 : ->s
combines    MONOIDE[s,*,1],
            DISTRIBUTIVITE[s,+,*]
```

$A_2 =$

```
property    ANNEAU2
sorts       s
operators   +, * : s, s ->s
            0, 1 : ->s
            i : s ->s
combines    MONOIDE-DISTRIBUTIF[s,+,*,1],
            GROUPE-COMM[s,+,0,i]
```

La combinaison des spécifications M et D peut être exprimée à l'aide d'une somme amalgamée, en partageant l'opérateur binaire * de M avec l'opérateur * de D. La spécification modulaire MD peut donc être exprimée de la façon suivante :

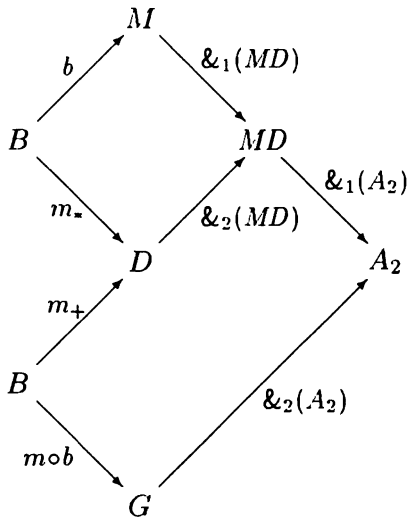
$MD = \mathsf{push}\,(B, M, D, b, m_*)$



On peut remarquer que dans cette expression, il est nécessaire d'avoir exprimé la spécification B pour l'opérateur binaire partagé, alors qu'en LPG, cette spécification partagée reste implicite. En effet, on n'a pas fait référence à `OP-BIN` dans la spécification `MONOIDE-DISTRIBUTIF`.

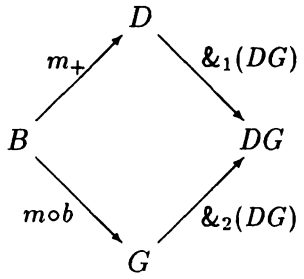
On peut ensuite exprimer A_2 comme la somme amalgamée de MD et G en partageant l'opérateur binaire + de MD avec l'opérateur binaire + de G.

$A_2 = \mathsf{push}\,(B, MD, G, \&_2(MD) \circ m_+, m \circ b)$

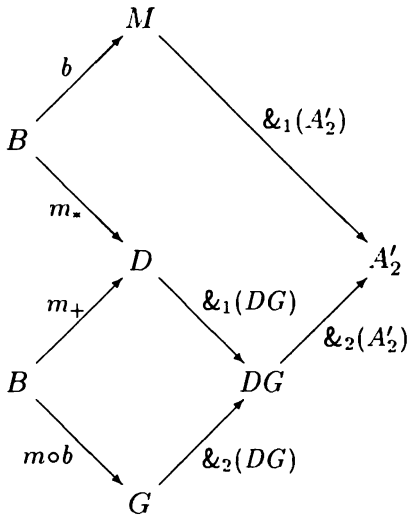


On peut effectuer ces deux combinaisons successives différemment, en combinant d'abord les spécifications G et D (en partageant l'opérateur +), puis en ajoutant la spécification M (en partageant l'opérateur *). Cela donne la spécification A_2'.

$DG = \mathsf{push}\,(B, D, G, m_+, m \circ b)$



$A'_2 = \mathsf{push}\,(B, M, DG, b, \&_1(DG) \circ m_*)$



On peut maintenant se poser la question suivante :

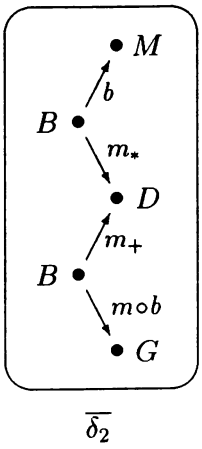
Avons-nous bien obtenu une spécification des anneaux ?

Dans ce cas particulier, on peut s'en convaincre assez facilement, en connaissant une définition indépendante des anneaux. Ici, on va s'intéresser à la question plus facile :

Est-ce que les spécifications A_2 et A'_2 sont équivalentes ?

Pour répondre à cette question, il est inutile d'avoir une définition indépendante des anneaux. On cherche simplement à savoir si les deux spécifications modulaires sont isomorphes. Ici, il est facile de voir que A_2 et A'_2 sont bien isomorphes. Intuitivement, les deux constructions sont équivalentes parce que l'opération de somme amalgamée est "associative" (dans un sens qui resterait à définir formellement). En fait, les deux

constructions A_2 et A'_2 sont un codage par des sommes amalgamées de la colimite du même diagramme $\overline{\delta_2}$ suivant.

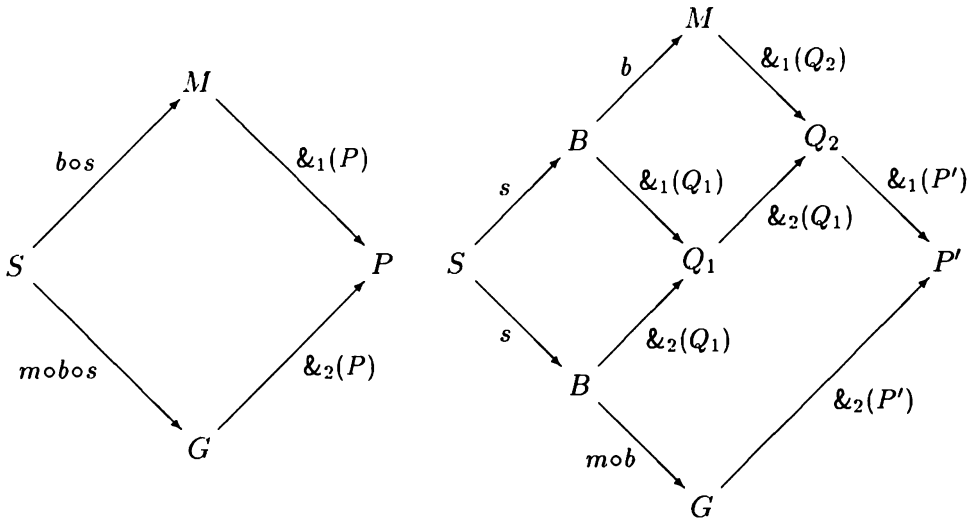


$\overline{\delta_2}$

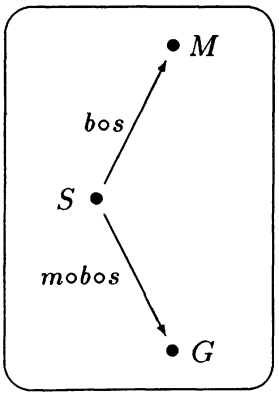
Mais des cas plus complexes peuvent se produire, comme le montre l'exemple suivant. Nous commençons par spécifier un pseudo-anneau, c'est-à-dire un anneau sans la propriété de distributivité. On peut faire cela par exemple avec le terme P, ou avec le terme P'.

$$P = \textsf{push}\ (S, M, G, b \circ s, m \circ b \circ s)$$

$$Q_1 = \textsf{push}\ (S, B, B, s, s)$$
$$Q_2 = \textsf{push}\ (B, M, Q_1, b, \&_1(Q_1))$$
$$P' = \textsf{push}\ (B, Q_2, G, \&_2(Q_2) \circ \&_2(Q_1), m \circ b)$$



En fait, P et P' sont deux spécifications isomorphes des pseudo-anneaux, qui correspondent à la colimite du diagramme $\overline{\alpha}$.

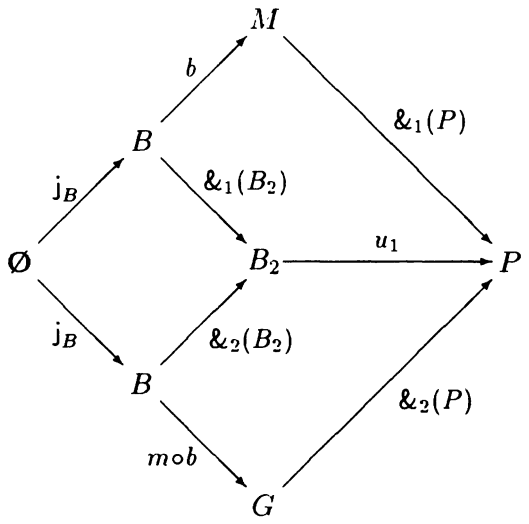


$\overline{\alpha}$

Nous allons maintenant "ajouter la distributivité" à la spécification P. Pour cela, nous considérons le morphisme de spécifications

$$u_1 = \mathsf{up}\,(\emptyset, B.B.P, \mathsf{j}_B, \mathsf{j}_B, \&_1(P) \circ b, \&_2(P) \circ m \circ b) : B_2 \to P$$

Cette flèche existe car comme $\emptyset$ est initial, $\&_1(P) \circ b \circ \mathsf{j}_B = \&_2(P) \circ m \circ b \circ \mathsf{j}_B$.



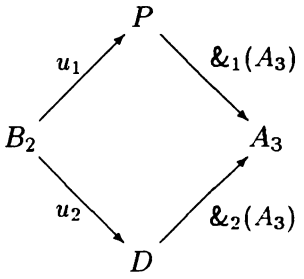
Nous avons vu en exemple 1.9 qu'il existe un morphisme de spécifications

$$u_2 = \mathsf{up}\,(\emptyset, B, B, D, \mathsf{j}_B, \mathsf{j}_B, m_*, m_+) : B_2 \to D$$

On obtient alors une spécification A_3 des anneaux en faisant la somme amalgamée de D et P par rapport à u_2 et u_1. Cela revient à partager les deux opérateurs binaires de P et D.

$$A_3 = \mathsf{push}\,(B_2, P, D, u_1, u_2)$$

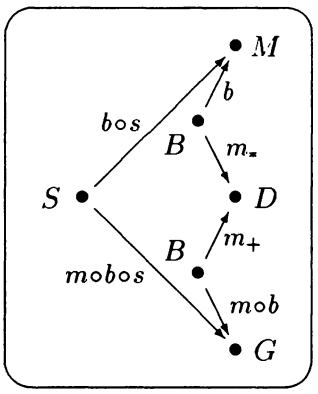
$$= \mathsf{push}\,(\mathsf{push}\,(\emptyset, B, B, \mathrm{j}_B, \mathrm{j}_B), P, D,$$
$$\mathsf{up}\,(\emptyset, B, B, P, \mathrm{j}_B, \mathrm{j}_B, \&_1(P) \circ b, \&_2(P) \circ m \circ b),$$
$$\mathsf{up}\,(\emptyset, B, B, D, \mathrm{j}_B, \mathrm{j}_B, m_*, m_+))$$



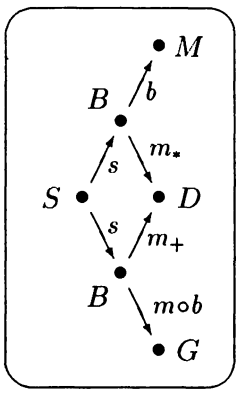
Enfin, voici une dernière spécification modulaire des anneaux, en utilisant la spécification P' des pseudo-anneaux à laquelle on ajoute la propriété de distributivité.

$$A_4 = \mathsf{push}\,(\mathsf{push}\,(\emptyset, B, B, \mathrm{j}_B, \mathrm{j}_B), P', D,$$
$$\mathsf{up}\,(\emptyset, B, B, P', \mathrm{j}_B, \mathrm{j}_B, \&_1(P') \circ \&_2(Q_2) \circ \&_1(Q_1),$$
$$\&_1(P') \circ \&_2(Q_2) \circ \&_2(Q_1)),$$
$$\mathsf{up}\,(\emptyset, B, B, D, \mathrm{j}_B, \mathrm{j}_B, m_*, m_+))$$

Les deux constructions modulaires d'anneaux A_3 et A_4 correspondent aux colimites des diagrammes $\overline{\delta_3}$ et $\overline{\delta_4}$ ci-dessous.



$\overline{\delta_3}$



$\overline{\delta_4}$

En utilisant la définition de colimite en théorie des catégorie, on peut vérifier que les *colimites* des diagrammes $\overline{\delta_1}$, $\overline{\delta_2}$, $\overline{\delta_3}$ et $\overline{\delta_4}$ sont isomorphes. Cet isomorphisme vient de l'égalité des morphismes de spécifications

$$m_+ \circ s = m_* \circ s.$$

Cette égalité signifie que le fait que les deux opérateurs + et * opèrent sur le même ensemble est contenu dans la spécification D des opérateurs distributifs.

Nous verrons dans le chapitre 2 que dans la catégorie de diagrammes $\mathrm{diagr}(\mathcal{C}_0)$, les diagrammes $\overline{\delta_1}$, $\overline{\delta_2}$, $\overline{\delta_3}$ et $\overline{\delta_4}$ sont isomorphes.

1.6 Isomorphisme de construction

Nous venons de voir qu'il existe plusieurs manières équivalentes de spécifier des anneaux à partir d'une catégorie de spécifications de base donnée. Dans ce paragraphe, nous examinons quelques équivalences possibles entre deux spécifications. Formellement, deux spécifications sont équivalentes si elles sont isomorphes dans une certaine catégorie de spécifications. Nous présentons les isomorphismes de construction, qui nous semblent présenter un intérêt particulier dans l'étude des spécifications modulaires.

Identité. Deux spécifications sont équivalentes si et seulement si celles-ci sont identiques. Il s'agit de la plus faible équivalence possible, qui n'est pas très intéressante.

Équivalence structurelle. Deux spécifications sont équivalentes si et seulement si celles-ci ont été construites de la même façon, indépendamment de noms intermédiaires qui ont pu être utilisés pendant la construction. Avec cette approche, on a la possibilité de nommer des spécifications modulaires intermédiaires et de réutiliser ces noms par la suite. Cette équivalence n'est pas beaucoup plus intéressante que l'identité.

Isomorphisme dans **Spec**. Deux spécifications sont isomorphes si et seulement si il existe un isomorphisme entre ces deux spécifications dans la catégorie des spécifications **Spec**. La difficulté ici est tout d'abord d'exhiber l'isomorphisme entre les deux signatures, et surtout de vérifier qu'il s'agit bien d'un isomorphisme de spécifications. Avec la définition de morphisme de spécifications donnée ici, il faut être capable de savoir si une équation appartient à la fermeture sémantique d'un ensemble d'équations. Ce n'est pas trivial puisqu'il faut pouvoir dire si une équation est satisfaite par une classe d'algèbres spécifiée par un ensemble d'équations. On peut pour cela utiliser un système logique, mais ce système n'est pas forcément complet. De plus, même si l'on dispose d'un système de déduction complet, le problème est, en général, uniquement semi-décidable.

Isomorphisme de construction. Dans tout notre travail, les isomorphismes que nous considérons sont des isomorphismes *de construction*. Deux spécifications sont isomorphes si on peut le prouver en utilisant les propriétés générales des colimites. Il s'agit d'un isomorphisme dans la catégorie librement engendrée par certaines colimites finies sur la catégorie des spécifications de base $\mathcal{C}_0$. L'intérêt de cet isomorphisme réside d'une part dans le fait qu'il n'est pas *trop général*, parce qu'il reflète la *construction* de la spécification modulaire ; et d'autre part dans le fait qu'il est suffisamment général pour faire abstraction des étapes particulières choisies pour la construction modulaire. Ces isomorphismes de construction ne dépendent pas de la définition effective des spécifications, mais seulement de la manière dont ces spécifications, liées par des morphismes de spécifications, sont combinées. En particulier, une modification "mineure" dans la catégorie des spécifications de base ne remettra pas en

cause des isomorphismes de construction. (Par "mineure" on entend ici que la modification ne change pas les égalités entre morphismes de spécifications de base existants). Par contre, si deux spécifications de base sont isomorphes et que cet isomorphisme ne fait pas partie de la catégorie des spécifications de base, alors celles-ci ne seront évidemment pas liées par un isomorphisme de construction.

1.7 Cadre général

Les spécifications que nous avons données au cours de ce chapitre sont des spécifications équationnelles. En effet, les seules formules utilisées sont des équations. Il existe évidemment des spécifications plus générales, par exemple des spécifications qui utilisent des clauses de Horn, ou la logique du premier ordre, ou encore les spécifications avec contraintes.

Pour formaliser la notion de système logique, J. Goguen et R. Burstall ont proposé les *institutions* [GB84, GB90]. Ce formalisme permet de parler de façon abstraite des signatures, des formules sur une signature, des modèles sur une signature et de la notion de satisfaction d'une formule par un modèle. L'intérêt des institutions est de fournir des résultats indépendants de la logique sous-jacente choisie.

Dans notre travail, nous étudions la composition des spécifications pour former des spécifications plus complexes à l'aide de colimites. Ce problème est identique qu'on se place par exemple dans le cadre des spécifications équationnelles, ou des spécifications avec clauses de Horn. En fait, on peut raisonner indépendamment de la logique sous-jacente en se plaçant dans le cadre général des institutions. La seule propriété que nous exigeons est l'existence de *colimites finies* dans la catégorie des spécifications.

1.7.1 Institutions

Rappelons d'abord la définition d'une *institution* [GB90].

Définition 1.17 (Institution) Une institution consiste en

- une catégorie **Sig**, dont les objets sont appelés *signatures*, et les flèches *morphismes de signatures*;
- un foncteur Sen : **Sig** $\rightarrow$ **Set** qui associe à chaque signature un ensemble de *propositions*, et à chaque morphisme de signatures une *application* entre deux ensembles de propositions ;
- un foncteur Mod : **Sig** $\rightarrow$ $\mathbf{Cat}^{\mathbf{op}}$ qui associe à chaque signature une catégorie de *modèles*, et à chaque morphisme de signatures un foncteur contravariant ;
- pour tout objet Σ de **Sig**, une relation $\models_\Sigma$ entre les objets de $Mod(\Sigma)$ et les éléments de $Sen(\Sigma)$, appelée relation de satisfaction, telle que pour tout

morphisme de signatures $m : \Sigma \to \Sigma'$, tout modèle M' objet de $Mod(\Sigma')$, et toute équation e de $Sen(\Sigma)$,

$$M' \models_{\Sigma'} Sen(m)(e) \Leftrightarrow Mod(m)(M') \models_{\Sigma} e. \qquad (i)$$

Les spécifications équationnelles que nous avons utilisées dans ce chapitre forment une institution.

- **Sig** est la catégorie des signatures définie en section 1.1.
- *Sen* associe à toute signature Σ l'ensemble des équations sur Σ, et à tout morphisme de signatures $m : \Sigma \to \Sigma'$ l'application $m^{\#}$ qui traduit les équations.
- *Mod* associe à toute signature Σ la catégorie des Σ-algèbres $\mathrm{Alg}(\Sigma)$, et à tout morphisme de signatures $m : \Sigma \to \Sigma'$ le *foncteur d'oubli*

  $$\mathcal{U}_m : \mathrm{Alg}(\Sigma') \to \mathrm{Alg}(\Sigma).$$

  Ce foncteur d'oubli permet de considérer toute Σ'-algèbre comme une Σ-algèbre.
- Soit une signature Σ, une Σ-algèbre A et une équation e sur Σ,

  $$A \models_{\Sigma} e \Leftrightarrow \text{l'algèbre } A \text{ satisfait l'équation } e \text{ (cf. définition 1.10).}$$

  La propriété (i) est vérifiée car pour toute algèbre A' de $\mathrm{Alg}(\Sigma')$,

  $$A' \models m^{\#}(e) \Leftrightarrow \mathcal{U}_m(A') \models e$$

  (cf. par exemple [GB90, EM90]).

Définition 1.18 (Spécification) Une spécification est un couple (Σ, E) où Σ est un objet de **Sig**, et E un sous-ensemble de $Sen(\Sigma)$.

C'est ce que J. Goguen et R. Burstall appellent une Σ-présentation [GB90].

Définition 1.19 (Fermeture sémantique d'un ensemble de propositions)
Soit une signature Σ et un ensemble de propositions $E \subseteq Sen(\Sigma)$. La *fermeture sémantique* $\overline{E} \subseteq Sen(\Sigma)$ de E est définie par :

$e \in \overline{E}$ si et seulement si pour tout modèle M de $Mod(\Sigma)$,

$$\forall e_0 \in E,\ M \models_{\Sigma} e_0 \Rightarrow M \models_{\Sigma} e.$$

On dit que E est fermé si et seulement si $E = \overline{E}$.

Définition 1.20 (Morphisme de spécifications) Soit deux spécifications (Σ, E) et (Σ', E'). Un *morphisme de spécifications* de (Σ, E) vers (Σ', E') est un morphisme de signatures $m : \Sigma \to \Sigma'$, tel que

$$\forall e \in E,\ Sen(m)(e) \in \overline{E'}.$$

Les spécifications, avec les morphismes de spécifications, forment une catégorie, notée **Spec**.

Notre but est de composer les spécifications par des constructions de colimites. Nous devons donc travailler dans une catégorie pour laquelle tout diagramme fini a une colimite, c'est-à-dire dans une catégorie finiment cocomplète. J. Goguen et R. Burstall ont montré qu'il suffit que la catégorie des signatures ait des colimites pour que la catégorie des spécifications associée ait des colimites.

Théorème 1.4 *([GB90]) La catégorie* **Spec** *est finiment cocomplète si et seulement si la catégorie associée de signatures* **Sig** *est finiment cocomplète.*

Remarque 1.4 J. Goguen et R. Burstall ont montré ce résultat pour les *théories*, c'est-à-dire les spécifications (Σ, E) telles que E est fermé. On vérifie facilement qu'on peut adapter leur démonstration dans le cas des spécifications.

Cela permet de déduire que la catégorie des spécifications algébriques équationnelles, la catégorie des spécifications du premier ordre ... etc. sont finiment cocomplètes.

1.7.2 Terminologie et hypothèses

Hypothèses

- Nous nous plaçons dans une institution, dans laquelle la catégorie des signatures **Sig** est finiment cocomplète. Cela implique que la catégorie des spécifications associée **Spec** est finiment cocomplète.
- On considère une sous-catégorie finie $\mathcal{C}_0$ de **Spec**.

Terminologie

Rappelons brièvement la terminologie employée.

Spécification et morphisme de spécifications. Une spécification et un morphisme de spécifications sont respectivement un objet et une flèche de la catégorie **Spec**.

Spécification de base et morphisme de spécifications de base. On fixe une sous-catégorie finie $\mathcal{C}_0$ de **Spec**. Les objets de $\mathcal{C}_0$ sont appelés *spécifications de base*, et les flèches de $\mathcal{C}_0$ *morphismes de spécifications de base*. Intuitivement, cette catégorie représente l'ensemble des spécifications et morphismes de spécifications "déclarés" par l'utilisateur, qui forment une bibliothèque de spécifications réutilisables.

Spécification modulaire et morphisme de spécifications modulaires. Une spécification modulaire est soit une spécification de base soit une spécification construite à partir d'autres spécifications modulaires par certaines constructions de colimites. Les morphismes de spécifications modulaires sont des morphismes de spécifications également obtenus par des constructions de colimites.

Angles d'études

Nous nous intéressons à la composition des spécifications de base sous différents angles :

Syntaxe. Peut-on définir une syntaxe, c'est-à-dire un langage pour décrire des constructions modulaires ? Dans le chapitre 3 nous proposons la construction d'une catégorie Terme($\mathcal{C}_0$). Cette catégorie permet de représenter la construction des spécifications modulaires à partir de spécifications de base en utilisant uniquement la spécification vide et des sommes amalgamées. Ce chapitre permet de justifier formellement la syntaxe abstraite utilisée dans l'exemple des anneaux.

Interprétation. Quel est le résultat de plusieurs constructions modulaires successives ? Nous montrons dans le chapitre 4 que des constructions successives de sommes amalgamées peuvent être vue comme le calcul d'une seule colimite. Nous montrons comment calculer le diagramme associé à un terme qui représente une spécification modulaire. Dans ce chapitre, une spécification modulaire est vue de façon plus abstraite comme la colimite d'un diagramme sur $\mathcal{C}_0$.

Calcul. Peut-on détecter des isomorphismes de construction ? Nous résolvons ce problème uniquement dans le cas simple où la catégorie $\mathcal{C}_0$ est finie et ne comporte pas de cycle. Nous proposons un algorithme qui permet, par exemple, de déduire que les spécifications A_1, A_2, A_3 et A_4 des anneaux que nous avons présentées dans ce chapitre sont équivalentes. Cet algorithme est présenté dans le chapitre 5.

Chapitre 2

Graphes, catégories et diagrammes

Le but de ce chapitre est de présenter les bases formelles de notre travail, en particulier les concepts de *diagramme* et de *morphisme de diagrammes*, issus de la théorie des catégories. Pour cela nous avons besoin de quelques notions sur les graphes, présentées en section 2.1. Nous introduisons la définition des *morphismes généralisés de graphes*. En section 2.2, nous faisons quelques rappels de théorie des catégories. La plupart des résultats sont tirés de [McL71] ou [MB70]. Sur ce sujet, il est également intéressant de consulter [BW90] qui introduit les concepts de base de façon assez claire et élémentaire. En section 2.3, nous présentons les notions de diagramme et de morphisme de diagrammes.

La définition de morphisme de diagrammes que nous utilisons, bien qu'elle ne soit pas nouvelle, est plus générale que celles présentées classiquement dans la littérature informatique (cf. par exemple [TBG91]). Nous avons en effet besoin d'une définition suffisamment générale pour "refléter" le maximum de flèches entre colimites de diagrammes par des morphismes entre les diagrammes correspondants. Nous nous basons ensuite sur cette définition pour reformuler les notions classiques de cône sur un diagramme et de colimite d'un diagramme.

Nous présentons deux catégories de diagrammes: DIAGR($\mathcal{C}$) et diagr($\mathcal{C}$). La catégorie DIAGR($\mathcal{C}$), qui est une généralisation de catégories de diagrammes utilisées en informatique, permet, de façon pratique, de manipuler les morphismes de diagrammes. La catégorie diagr($\mathcal{C}$), obtenue comme un quotient de DIAGR($\mathcal{C}$) par une relation de congruence sur les flèches de DIAGR($\mathcal{C}$), possède deux propriétés théoriques importantes. D'une part, diagr($\mathcal{C}$) est une catégorie finiment cocomplète (théorème 2.6). De plus, diagr($\mathcal{C}$) est une complétion de la catégorie $\mathcal{C}$ par colimites finies (théorème 2.7).

2.1 Graphes

Définition 2.1 (Graphe) Un graphe α^Φ est un ensemble de nœuds Nœud(α^Φ), un ensemble d'arcs Arc(α^Φ), et deux fonctions Source, But : Arc(α^Φ) $\rightarrow$ Nœud(α^Φ). Si a est un arc de source m et de but n, on note $a : m \rightarrow n$.

Un graphe est *fini* lorsque l'ensemble des nœuds et l'ensemble des arcs sont finis.

Définition 2.2 (Morphisme de graphes) Soit deux graphes α^Φ et β^Φ. Un morphisme de graphes σ^Φ de α^Φ vers β^Φ, noté $\sigma^\Phi : \alpha^\Phi \to \beta^\Phi$, est une application qui à tout nœud n de α^Φ associe un nœud $\sigma^\Phi(n)$ de β^Φ et à tout arc $a : m \to n$ de α^Φ associe un arc $\sigma^\Phi(a) : \sigma^\Phi(m) \to \sigma^\Phi(n)$ de β^Φ.

Composition des morphismes de graphes. Soit trois graphes α^Φ, β^Φ, γ^Φ, et deux morphismes de graphes $\sigma^\Phi : \alpha^\Phi \to \beta^\Phi$ et $\tau^\Phi : \beta^\Phi \to \gamma^\Phi$. La composition de σ^Φ et τ^Φ est le morphisme $\tau^\Phi \circ \sigma^\Phi : \alpha^\Phi \to \gamma^\Phi$ qui associe à tout nœud n de α^Φ le nœud $\tau^\Phi(\sigma^\Phi(n))$ de γ^Φ, et à tout arc $a : m \to n$ de α^Φ associe l'arc $\tau^\Phi(\sigma^\Phi(a)) : \tau^\Phi(\sigma^\Phi(m)) \to \tau^\Phi(\sigma^\Phi(n))$. En résumé, $\tau^\Phi \circ \sigma^\Phi$ est défini de la façon suivante.

$$\begin{array}{rccl} \tau^\Phi \circ \sigma^\Phi & : & \alpha^\Phi & \to & \gamma^\Phi \\ & & n & \mapsto & \tau^\Phi(\sigma^\Phi(n)) \\ & & a : m \to n & \mapsto & \tau^\Phi(\sigma^\Phi(a)) : \tau^\Phi(\sigma^\Phi(m)) \to \tau^\Phi(\sigma^\Phi(n)) \end{array}$$

Définition 2.3 (Zigzag sur un graphe) Soit un graphe α^Φ. Un *zigzag* Z sur α^Φ est un triplet (k, Z_N, Z_A) où

- k est un entier naturel, appelé *longueur* du zigzag Z ;
- Z_N est un $(k+1)$-uplet $(n_0, n_1, \ldots, n_k)$ de nœuds de α^Φ ;
- Z_A est un k-uplet $(a_0, a_1, \ldots, a_{k-1})$ d'arcs de α^Φ, tel que pour tout entier naturel i tel que $0 \leq i \leq k-1$, a_i est soit un arc de source n_i et de but n_{i+1}, soit un arc de source n_{i+1} et de but n_i.

On appelle n_0 la *source* du zigzag Z, et n_k son *but*. On note $Z : n_0 \rightsquigarrow n_k$.
Pour tout nœud n de α^Φ, on a un *zigzag nul* de longueur nulle noté $0_n : n \rightsquigarrow n$.

Par la suite, pour plus de lisibilité, un zigzag $Z : n_0 \rightsquigarrow n_k$ sera représenté de la façon suivante :

$$Z = n_0 \xrightarrow{a_0} n_1 \xleftarrow{a_1} n_2 \xleftarrow{a_2} n_3 \cdots n_{k-1} \xrightarrow{a_{k-1}} n_k.$$

On choisit une orientation arbitraire pour la représentation de chaque arc a_i.

Étant donné un graphe α^Φ, on a un graphe Zigzag(α^Φ), dont les nœuds sont les nœuds de α^Φ, et les arcs les zigzags de α^Φ.

À tout zigzag $Z : n_0 \rightsquigarrow n_k$

$$Z = n_0 \xrightarrow{a_0} n_1 \xleftarrow{a_1} n_2 \xleftarrow{a_2} n_3 \cdots n_{k-1} \xrightarrow{a_{k-1}} n_k,$$

on peut associer un *zigzag opposé* $\overline{Z} : n_k \rightsquigarrow n_0$

$$\overline{Z} = n_k \xleftarrow{a_{k-1}} n_{k-1} \cdots n_3 \xrightarrow{a_2} n_2 \xrightarrow{a_1} n_1 \xleftarrow{a_0} n_0$$

obtenu en "parcourant le zigzag dans l'autre sens". Plus formellement, on définit l'opposé d'un zigzag de la façon suivante.

Définition 2.4 (Zigzag opposé) Soit un zigzag $Z = (k, Z_N, Z_A)$ sur un graphe α^Φ, avec $Z_N = (n_0, n_1, \ldots, n_k)$ et $Z_A = (a_0, a_1, \ldots, a_{k-1})$. L'opposé du zigzag

$$Z : n_0 \rightsquigarrow n_k$$

est le zigzag

$$\overline{Z} : n_k \rightsquigarrow n_0 = (k, Z'_N, Z'_A)$$

où

- $Z'_N = (n_k, n_{k-1}, \ldots, n_0)$;
- $Z'_A = (a_{k-1}, \ldots, a_0)$.

Définition 2.5 (Morphisme généralisé de graphes) Soit deux graphes α^Φ et β^Φ. Un *morphisme généralisé de graphes* $\sigma^\Phi : \alpha^\Phi \rightsquigarrow \beta^\Phi$ entre les graphes α^Φ et β^Φ est un morphisme de graphes entre α^Φ et Zigzag(β^Φ).

Le morphisme généralisé de graphes σ^Φ associe donc à tout nœud n de α^Φ un nœud $\sigma^\Phi(n)$ de β^Φ, et à tout arc $a : m \to n$ de α^Φ un *zigzag* $\sigma^\Phi(a) : \sigma^\Phi(m) \rightsquigarrow \sigma^\Phi(n)$ du graphe β^Φ.

Tout morphisme de graphes est donc un morphisme généralisé, puisqu'un arc est un cas particulier de zigzag.

Un morphisme généralisé associe donc à tout arc un zigzag, en conservant les sources et les buts. On peut étendre la définition d'un morphisme généralisé de graphes $\sigma^\Phi : \alpha^\Phi \rightsquigarrow \beta^\Phi$ en associant à tout zigzag un zigzag. Il suffit pour cela de "mettre les zigzags bout à bout". Étant donné un zigzag $Z : n_0 \rightsquigarrow n_k$

$$Z \;=\; n_0 \xleftarrow{a_0} n_1 \xrightarrow{a_1} n_2 \cdots n_{k-1} \xrightarrow{a_{k-1}} n_k,$$

on définit le zigzag $\sigma^\Phi(Z) : \sigma^\Phi(n_0) \rightsquigarrow \sigma^\Phi(n_k)$ par :

$$\sigma^\Phi(Z) \;=\; \sigma^\Phi(n_0) \overset{\sigma^\Phi(a_0)}{\leftsquigarrow} \sigma^\Phi(n_1) \overset{\sigma^\Phi(a_1)}{\rightsquigarrow} \sigma^\Phi(n_2) \cdots \sigma^\Phi(n_{k-1}) \overset{\sigma^\Phi(a_{k-1})}{\rightsquigarrow} \sigma^\Phi(n_k).$$

Cela permet de composer les morphismes généralisés. Soit trois graphes α^Φ, β^Φ, γ^Φ, et deux morphismes généralisés $\sigma^\Phi : \alpha^\Phi \rightsquigarrow \beta^\Phi$, $\tau^\Phi : \beta^\Phi \rightsquigarrow \gamma^\Phi$. Le morphisme généralisé $\tau^\Phi \circ \sigma^\Phi$ est défini de la façon suivante.

$$\begin{array}{rccl} \tau^\Phi \circ \sigma^\Phi \;:\; & \alpha^\Phi & \to & \gamma^\Phi \\ & n & \mapsto & \tau^\Phi(\sigma^\Phi(n)) \\ & a : m \to n & \mapsto & \tau^\Phi(\sigma^\Phi(a)) : \tau^\Phi(\sigma^\Phi(m)) \rightsquigarrow \tau^\Phi(\sigma^\Phi(n)) \end{array}$$

2.2 Catégories, foncteurs ...

Dans cette partie, nous rappelons les définitions de catégorie, foncteur, transformation naturelle et adjonction. Ces définitions sont tirées de [McL71, MB70, BW90].

2.2.1 Catégories

Définition 2.6 (Catégorie) Une catégorie $\mathcal{C}$ est définie de la façon suivante.

- On a une classe d'*objets*, notés A, B, C ...
- Pour chaque couple d'objets (A, B), on a un ensemble $hom_{\mathcal{C}}(A, B)$ de *flèches*, noté plus simplement $hom(A, B)$ s'il n'y pas d'ambiguïté sur la catégorie.

  $f \in hom(A, B)$ est noté $f : A \to B$.

  A est appelé *domaine* de f, et B est appelé *codomaine* de f.
- Pour chaque triplet d'objets (A, B, C), on a une opération de composition

$$\begin{array}{rccc} \circ & : & hom(B,C) \times hom(A,B) & \to & hom(A,C) \\ & & (g, f) & \mapsto & g \circ f \end{array}$$

- Pour tout objet B, on a une flèche $id_B : B \to B$.
- Pour tout objet B, la flèche id_B est une *identité*:

  $\forall f : A \to B,\ \forall g : B \to C,\ id_B \circ f = f$ et $g \circ id_B = g$.
- La composition est associative:

  $\forall f : A \to B,\ \forall g : B \to C,\ \forall h : C \to D,\ h \circ (g \circ f) = (h \circ g) \circ f$.

Exemples de catégories

- La catégorie des ensembles, notée **Set**, dont les objets sont les ensembles, et les flèches sont les applications entre deux ensembles.
- La catégorie des signatures, notée **Sig**, dont les objets sont les signatures, et les flèches sont les morphismes de signatures.
- La catégorie des spécifications, notée **Spec**, dont les objets sont les spécifications algébriques, et les flèches sont les morphismes de spécifications.
- La catégorie des graphes, notée **Graph**, dont les objets sont les graphes, et les flèches sont les morphismes de graphes.

Définition 2.7 (Isomorphisme) Soit une catégorie $\mathcal{C}$ et $f : A \to B$ une flèche de $\mathcal{C}$. On dit que $f : A \to B$ est un *isomorphisme* si et seulement si il existe une flèche $g : B \to A$ de $\mathcal{C}$ telle que $g \circ f = id_A$ et $f \circ g = id_B$.
Si f est un isomorphisme, la flèche g est alors unique, et est notée f^{-1}. On dit alors que f et f^{-1} sont *inverses* et que les objets A et B sont *isomorphes*. On note $A \cong B$.

Définition 2.8 (Petite catégorie) Lorsque la classe d'objets d'une catégorie $\mathcal{C}$ est un *ensemble*, on dit que $\mathcal{C}$ est une *petite catégorie*.

Par exemple, les catégories **Set**, **Sig**, **Spec** et **Graph** ne sont pas des petites catégories.

Remarque 2.1

- Toute petite catégorie est un graphe.
- Réciproquement, on peut associer à tout graphe α^{Φ} la catégorie *librement engendrée* sur ce graphe. Cette catégorie a pour objets les nœuds du graphe α^{Φ}, et pour flèches les *chaînes d'arcs composables* de α^{Φ}, où deux arcs a et a' sont composables si et seulement si But(a) = Source(a'). La composition est définie comme la concaténation des chaînes: $(b_1 b_2 \dots b_l) \circ (a_1 a_2 \dots a_k) = (a_1 a_2 \dots a_k b_1 b_2 \dots b_l)$. Les flèches identités sur un objet n sont les chaînes de longueur nulle de source et but n, notées ε_n.

2.2.2 Foncteurs

Un foncteur est l'équivalent d'un morphisme de graphes pour les catégories.

Définition 2.9 (Foncteur) Soit $\mathcal{C}$ et $\mathcal{D}$ deux catégories. Un *foncteur* F de $\mathcal{C}$ vers $\mathcal{D}$, noté $F : \mathcal{C} \to \mathcal{D}$, est une application qui à tout objet A de $\mathcal{C}$ associe un objet $F(A)$ de $\mathcal{D}$, et à toute flèche $f : A \to B$ de $\mathcal{C}$ associe une flèche $F(f) : F(A) \to F(B)$ de $\mathcal{D}$, et telle que

- pour tout objet A de $\mathcal{C}$, $F(id_A) = id_{F(A)}$;
- pour toutes flèches $f : A \to B$ et $g : B \to C$ de $\mathcal{C}$, $F(g \circ f) = F(g) \circ F(f)$.

Étant donné une catégorie $\mathcal{C}$, il y a foncteur identité $Id_{\mathcal{C}} : \mathcal{C} \to \mathcal{C}$, qui est l'identité sur les objets et sur les flèches.
Soit trois catégories $\mathcal{C}$, $\mathcal{D}$ et $\mathcal{E}$, et deux foncteurs $F : \mathcal{C} \to \mathcal{D}$ et $G : \mathcal{D} \to \mathcal{E}$.
On définit la composition de F et G de façon évidente:

$$\begin{array}{rccl} G \circ F \ : & \mathcal{C} & \to & \mathcal{E} \\ & A & \mapsto & G(F(A)) \\ & f : A \to B & \mapsto & G(F(f)) \ : \ G(F(A)) \to G(F(B)) \end{array}$$

Catégorie Cat

La classe des petites catégories, avec les foncteurs, forme une catégorie **Cat**.

Catégorie CAT

Par la suite, nous aurons besoin d'une catégorie qui contient non seulement des petites catégories mais également des catégories comme **Set** ou **Spec** qui ne sont pas petites. Nous considérons pour cela la catégorie **CAT** des "grandes catégories", qui contient les petites catégories, ainsi que les autres catégories qui nous intéressent. Pour des problèmes de fondations, de même qu'on ne peut pas considérer l'ensemble de tous les ensembles, **CAT** n'est pas la catégorie de toutes les catégories. En particulier **CAT** n'est pas un objet de **CAT**. Pour plus de détails, on peut consulter [McL71] pages 21–24.

Propriété 2.1 *Soit un foncteur $F : \mathcal{C} \to \mathcal{D}$. L'image par F d'un isomorphisme de $\mathcal{C}$ est un isomorphisme de $\mathcal{D}$.*

Preuve. Soit un foncteur $F : \mathcal{C} \to \mathcal{D}$, et $f : A \to B$ un isomorphisme de $\mathcal{C}$. $F(f) : F(A) \to F(B)$ est un isomorphisme, et $F(f)^{-1} = F(f^{-1}) : F(B) \to F(A)$. En effet,

$$\begin{aligned} & F(f) \circ F(f^{-1}) \\ = \; & F(f \circ f^{-1}) && (F \text{ est un foncteur}) \\ = \; & F(id_B) && (\text{définition de } f^{-1}) \\ = \; & id_{F(B)} && (F \text{ est un foncteur}) \end{aligned}$$

De même, $F(f^{-1}) \circ F(f) = id_{F(A)}$, et donc $F(f)$ est un isomorphisme. □

Définition 2.10 (Endofoncteur) Un foncteur $F : \mathcal{C} \to \mathcal{C}$ est appelé *endofoncteur.*

Si F est un endofoncteur, on pose

$$\begin{aligned} & F^0 = Id_{\mathcal{C}} ; \\ & F^i = F \circ F^{i-1}, \quad \forall i \geq 1. \end{aligned}$$

Définition 2.11 (Foncteur plein) Soit deux catégories $\mathcal{C}$ et $\mathcal{D}$, et un foncteur $F : \mathcal{C} \to \mathcal{D}$. On dit que F est *plein* si pour tout couple d'objets (A, B) de $\mathcal{C}$, et pour toute flèche $g : F(A) \to F(B)$ de $\mathcal{D}$, il existe une flèche $f : A \to B$ de $\mathcal{C}$ telle que $g = F(f)$.

Intuitivement, entre deux objets de $\mathcal{D}$ images par F d'objets de $\mathcal{C}$, il n'y a pas de flèches autres que les images par F de flèches de $\mathcal{C}$. F est, d'une certaine façon, "surjectif sur les flèches".

Définition 2.12 (Foncteur fidèle) Soit deux catégories $\mathcal{C}$ et $\mathcal{D}$, et un foncteur $F : \mathcal{C} \to \mathcal{D}$. On dit que F est *fidèle* si pour tout couple d'objets (A, B) de $\mathcal{C}$, et pour tout couple de flèches $f, f' : A \to B$, on a $F(f) = F(f') \Rightarrow f = f'$.

Autrement dit, F est fidèle si F est injectif sur les flèches.

2.2.3 Transformations naturelles

Définition 2.13 (Transformation naturelle) Soit deux catégories $\mathcal{C}$ et $\mathcal{D}$. Soit deux foncteurs $F, G : \mathcal{C} \to \mathcal{D}$. Une *transformation naturelle* σ de F vers G, notée $\sigma : F \dot{\to} G$, est une application qui à tout objet A de $\mathcal{C}$ associe une flèche $\sigma_A : F(A) \to G(A)$ de $\mathcal{D}$, telle que pour toute flèche $f : A \to B$ de $\mathcal{C}$,

$$G(f) \circ \sigma_A = \sigma_B \circ F(f).$$

$$\begin{array}{ccc} A & & F(A) \xrightarrow{\sigma_A} G(A) \\ \Big\downarrow f & & \Big\downarrow F(f) \qquad\quad \Big\downarrow G(f) \\ B & & F(B) \xrightarrow[\sigma_B]{} G(B) \end{array}$$

Catégorie des foncteurs de $\mathcal{C}$ vers $\mathcal{D}$

Soit trois foncteurs $F, G, H : \mathcal{C} \to \mathcal{D}$, et deux transformations naturelles $\sigma : F \dot{\to} G$ et $\tau : G \dot{\to} H$. La composition de σ et τ est la transformation naturelle $\tau \cdot \sigma : F \dot{\to} H$ telle que pour tout objet A de $\mathcal{C}$,

$$(\tau \cdot \sigma)_A = \tau_A \circ \sigma_A$$

Cela définit bien une transformation naturelle. En effet,

$$\begin{array}{lll} H(f) \circ (\tau \cdot \sigma)_A & & \\ = & H(f) \circ \tau_A \circ \sigma_A & \text{(définition de } \tau \cdot \sigma) \\ = & \tau_B \circ G(f) \circ \sigma_A & (\tau \text{ transformation naturelle}) \\ = & \tau_B \circ \sigma_B \circ F(f) & (\sigma \text{ transformation naturelle}) \\ = & (\tau \cdot \sigma)_B \circ F(f) & \text{(définition de } \tau \cdot \sigma) \end{array}$$

$$\begin{array}{ccccc} A & & F(A) & \xrightarrow{\sigma_A} & G(A) \xrightarrow{\tau_A} H(A) \\ \Big\downarrow f & & \Big\downarrow F(f) & & \Big\downarrow G(f) \qquad \Big\downarrow H(f) \\ B & & F(B) & \xrightarrow[\sigma_B]{} & G(B) \xrightarrow[\tau_B]{} H(B) \end{array}$$

Remarque 2.2 Cette composition des transformations naturelles est souvent appelée composition *verticale*, par opposition à la composition *horizontale* dont nous n'avons pas rappelé la définition ici.

Soit un foncteur $F : \mathcal{C} \to \mathcal{D}$. On a une transformation naturelle identité, notée $id_F : F \dot{\to} F$, qui associe à tout objet A de $\mathcal{C}$ la flèche $id_{F(A)}$. On vérifie facilement que pour tous foncteurs $F, G : \mathcal{C} \to \mathcal{D}$, et toute transformation naturelle $\sigma : F \dot{\to} G$, on a

$$\sigma \cdot id_F = \sigma ;$$
$$id_G \cdot \sigma = \sigma.$$

On vérifie facilement que la composition des transformations naturelles est associative, donc on a une catégorie, notée $\mathcal{D}^{\mathcal{C}}$, dont les objets sont les foncteurs de $\mathcal{C}$ vers $\mathcal{D}$, et les flèches sont les transformations naturelles.

Définition 2.14 (Isomorphisme naturel) Soit deux catégories $\mathcal{C}$ et $\mathcal{D}$. Soit deux foncteurs $F, G : \mathcal{C} \to \mathcal{D}$. Un isomorphisme $\sigma : F \dot{\to} G$ de la catégorie $\mathcal{D}^{\mathcal{C}}$ est appelé *isomorphisme naturel*. Si $\sigma : F \dot{\to} G$ est un isomorphisme naturel, on dit que les foncteurs F et G sont *naturellement isomorphes*, et on note $F \cong G$.

Propriété 2.2 *Soit deux catégories $\mathcal{C}$ et $\mathcal{D}$. Soit deux foncteurs $F, G : \mathcal{C} \to \mathcal{D}$. Une transformation naturelle $\sigma : F \dot{\to} G$ est* un isomorphisme naturel *si et seulement si pour tout objet A de $\mathcal{C}$, $\sigma_A : F(A) \to G(A)$ est un isomorphisme.*
De plus, $\sigma^{-1} : G \dot{\to} F$ est la transformation naturelle qui associe à tout objet A de $\mathcal{C}$ la flèche $\sigma_A^{-1} : G(A) \to F(A)$ de $\mathcal{D}$.

Preuve. Notons $\tau_A = \sigma_A^{-1}$. On vérifie d'abord que τ est une transformation naturelle.

$$\begin{array}{lll} G(f) \circ \sigma_A = \sigma_B \circ F(f) & & (\sigma \text{ transformation naturelle}) \\ \Rightarrow \quad \sigma_B^{-1} \circ G(f) = F(f) \circ \sigma_A^{-1} & & (\sigma_A \text{ et } \sigma_B \text{ isomorphismes}) \\ \Rightarrow \quad \tau_B \circ G(f) = F(f) \circ \tau_A & & (\text{définition de } \tau_A \text{ et } \tau_B) \end{array}$$

donc $\tau : G \dot{\to} F$ est une transformation naturelle.

Montrons maintenant que σ et τ sont inverses. Pour tout objet A de $\mathcal{C}$,

$$(\sigma \cdot \tau)_A = \sigma_A \circ \tau_A = \sigma_A \circ \sigma_A^{-1} = id_{G(A)}$$

donc $\sigma \cdot \tau = id_G$. De même, on montre que $\tau \cdot \sigma = id_F$. Par conséquent, $\sigma^{-1} = \tau$. □

Considérons quatre catégories $\mathcal{B}$, $\mathcal{C}$, $\mathcal{D}$ et $\mathcal{E}$: et quatre foncteurs $K : \mathcal{B} \to \mathcal{C}$, $F, G : \mathcal{C} \to \mathcal{D}$, et $H : \mathcal{D} \to \mathcal{E}$. Considérons une transformation naturelle $\sigma : F \dot{\to} G$.

On a une transformation naturelle

$$H\sigma : H \circ F \dot{\to} H \circ G$$

telle que pour tout objet A de $\mathcal{C}$,

$$(H\sigma)_A = H(\sigma_A) : H(F(A)) \to H(G(A)) \qquad (\text{flèche de } \mathcal{E})$$

Cela définit bien une transformation naturelle. Soit une flèche $f : A \to B$ de $\mathcal{C}$.

$$\begin{array}{lll} G(f) \circ \sigma_A = \sigma_B \circ F(f) & & (\sigma \text{ transformation naturelle}) \\ \Rightarrow \quad H(G(f) \circ \sigma_A) = H(\sigma_B \circ F(f)) & & \\ \Rightarrow \quad H(G(f)) \circ H(\sigma_A) = H(\sigma_B) \circ H(F(f)) & & (H \text{ foncteur}) \\ \Rightarrow \quad (H \circ G)(f) \circ (H\sigma)_A = (H\sigma)_B \circ (H \circ F)(f) & & (\text{définition de } H\sigma) \end{array}$$

On a également une transformation naturelle

$$\sigma K : F \circ K \to G \circ K$$

telle que pour tout objet A de $\mathcal{B}$,

$$(\sigma K)_A = \sigma_{K(A)} : F(K(A)) \to G(K(A)) \qquad (\text{flèche de } \mathcal{D})$$

Cela définit bien une transformation naturelle. En effet, soit une flèche $f : A \to B$ de $\mathcal{B}$.

$$\begin{array}{lll} G(K(f)) \circ \sigma_{K(A)} = \sigma_{K(B)} \circ F(K(f)) & & (\sigma \text{ transformation naturelle}) \\ \Rightarrow \quad (G \circ K)(f) \circ (\sigma K)_A = (\sigma K)_B \circ (F \circ K)(f) & & (\text{définition de } \sigma K) \end{array}$$

Propriété 2.3 *Soit quatre catégories $\mathcal{B}$, $\mathcal{C}$, $\mathcal{D}$ et $\mathcal{E}$; et quatre foncteurs $K : \mathcal{B} \to \mathcal{C}$, $F, G : \mathcal{C} \to \mathcal{D}$, et $H : \mathcal{D} \to \mathcal{E}$. Supposons $F \cong G$. Alors,*

1. $H \circ F \cong H \circ G$;
2. $F \circ K \cong G \circ K$.

Preuve. Soit $\sigma : F \dot{\to} G$ l'isomorphisme naturel entre F et G.

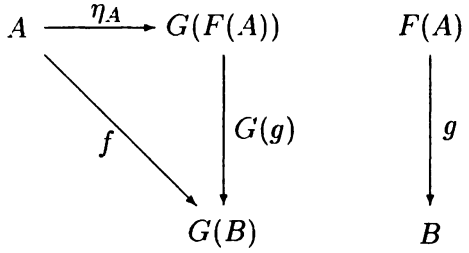
1. Pour tout objet A de $\mathcal{C}$, σ_A est un isomorphisme, donc, d'après la propriété 2.1, $H(\sigma_A) = (H\sigma)_A$ est un isomorphisme. Par conséquent, d'après la propriété 2.2, $H\sigma : H \circ F \dot{\to} H \circ G$ est un isomorphisme naturel, et donc $H \circ F \cong H \circ G$.
2. Pour tout objet A de $\mathcal{B}$, $\sigma_{K(A)} = (\sigma K)_A$ est un isomorphisme. Par conséquent, $\sigma K : F \circ K \dot{\to} G \circ K$ est un isomorphisme naturel, et donc $F \circ K \cong G \circ K$.

□

2.2.4 Adjonctions

Un des concepts les plus importants de la théorie des catégories est l'adjonction. Par exemple les constructions *libres* (comme les constructions d'*algèbres libres*, de *groupes libres* ou de *monoïdes libres*) sont caractérisées par des adjonctions.

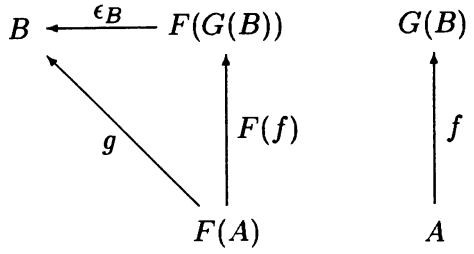
Définition 2.15 (Adjonction) Soit deux catégories $\mathcal{C}$ et $\mathcal{D}$. Soit deux foncteurs $F : \mathcal{C} \to \mathcal{D}$ et $G : \mathcal{D} \to \mathcal{C}$. Les foncteurs F et G forment une *adjonction* notée $(F \dashv G)$ si et seulement si il existe une transformation naturelle $\eta : Id_{\mathcal{C}} \dot{\to} G \circ F$ telle que pour tout objet A de $\mathcal{C}$, pour tout objet B de $\mathcal{D}$, et pour toute flèche $f : A \to G(B)$ de $\mathcal{C}$, il existe une unique flèche $g : F(A) \to B$ de $\mathcal{D}$ telle que $G(g) \circ \eta_A = f$.



La transformation naturelle $\eta : Id_{\mathcal{C}} \dot{\to} G \circ F$ est appelée *unité* de l'adjonction $(F \dashv G)$.

Théorème 2.1 *Soit deux catégories $\mathcal{C}$ et $\mathcal{D}$, et deux foncteurs $F : \mathcal{C} \to \mathcal{D}$ et $G : \mathcal{D} \to \mathcal{C}$. Alors, $(F \dashv G)$ est une adjonction si et seulement si il existe une transformation naturelle $\epsilon : F \circ G \dot{\to} Id_{\mathcal{D}}$ telle que pour tout objet B de $\mathcal{D}$, pour tout*

objet A de $\mathcal{C}$ et pour toute flèche $g : F(A) \to B$ de $\mathcal{D}$, il existe une unique flèche $f : A \to G(B)$ de $\mathcal{C}$ telle que $\epsilon_B \circ F(f) = g$.



La transformation naturelle $\epsilon : F \circ G \dot{\to} Id_{\mathcal{D}}$ est appelée coünité *de l'adjonction $(F \dashv G)$.*

Preuve. On peut trouver une preuve de ce théorème dans [BW90], page 275. □

Propriété 2.4 *Soit une adjonction $(F \dashv G)$, ayant pour unité la transformation naturelle $\eta : Id_{\mathcal{C}} \dot{\to} G \circ F$ et pour coünité la transformation naturelle $\epsilon : F \circ G \dot{\to} Id_{\mathcal{D}}$. Alors, on a les égalités suivantes entre transformations naturelles :*

1. $G\epsilon \cdot \eta G = id_G$:
2. $\epsilon F \cdot F\eta = id_F$.

Preuve. On peut trouver une preuve de cette propriété dans [McL71], page 80. □

Exemple 2.1 (Adjonction entre la catégorie **Graph** et la catégorie **Cat**)

À toute petite catégorie $\mathcal{C}$, on peut associer un graphe $U(\mathcal{C})$, dont les nœuds sont les objets de $\mathcal{C}$, et les arcs les flèches de $\mathcal{C}$. Réciproquement, à tout graphe α^Φ, on peut associer la catégorie $F(\alpha^\Phi)$ librement engendrée sur le graphe α^Φ (cf. remarque 2.1).

F et U sont en fait des foncteurs

$$F : \mathbf{Graph} \to \mathbf{Cat}$$
$$U : \mathbf{Cat} \to \mathbf{Graph}$$

qui forment une adjonction $(F \dashv U)$. En effet, pour toute catégorie $\mathcal{C}$, pour tout graphe α^Φ, pour tout morphisme de graphes $m : \alpha^\Phi \to U(\mathcal{C})$, il existe une unique extension de m en un foncteur $M : F(\alpha^\Phi) \to \mathcal{C}$.

Par abus de notation, dans toute la suite la catégorie librement engendrée sur un graphe α^Φ sera également notée α^Φ.

2.3 Diagrammes

Dans cette section, nous définissons les diagrammes sur une catégorie de base $\mathcal{C}$. Intuitivement, un diagramme décrit un assemblage d'objets de $\mathcal{C}$ à l'aide de flèches de $\mathcal{C}$. Nous rappelons la définition de la *colimite* d'un diagramme, qui peut être vue intuitivement comme le résultat de l'assemblage décrit par le diagramme.

Nous avons besoin de définir non seulement des diagrammes sur $\mathcal{C}$, mais également des morphismes de diagrammes, afin d'obtenir une *catégorie* de diagrammes. En effet, l'idée principale de notre travail est d'associer à toute spécification un diagramme, et à tout morphisme de spécifications un morphisme de diagrammes.

Dans cette section, nous commençons par définir la catégorie DIAGR($\mathcal{C}$), qui a pour objets les *diagrammes* et pour flèches les *morphismes de diagrammes*. Notre définition de morphisme de diagrammes, basée sur les notions de *zigzag* sur un graphe et de *morphisme généralisé de graphe* est plus générale que celle par exemple donnée dans [TBG91]. Puis, nous reformulons la définition de cône et de cône colimite.

Nous définissons ensuite une congruence $\approx$ sur les flèches de DIAGR($\mathcal{C}$) et la catégorie diagr($\mathcal{C}$) est le quotient de DIAGR($\mathcal{C}_0$) par cette relation. La catégorie diagr($\mathcal{C}_0$) a pour objets les objets de DIAGR($\mathcal{C}$), et pour flèches les classes d'équivalence modulo $\approx$ de flèches de DIAGR($\mathcal{C}$). Nous montrons que la catégorie diagr($\mathcal{C}$) est finiment cocomplète et que le calcul d'une colimite consiste à *aplatir* un diagramme de diagrammes dans la catégorie DIAGR($\mathcal{C}$). Nous montrons enfin que diagr($\mathcal{C}$) est une complétion de $\mathcal{C}$ par colimites finies.

2.3.1 Catégorie des diagrammes DIAGR($\mathcal{C}$)

Un diagramme $\overline{\alpha}$ sur une catégorie $\mathcal{C}$ est un graphe α^Φ, dont les nœuds n sont étiquetés par des objets $\alpha(n)$ de $\mathcal{C}$, et les arcs $a : n \to n'$ sont étiquetés par des flèches $\alpha(a) : \alpha(n) \to \alpha(n')$ de $\mathcal{C}$. De manière équivalente :

Définition 2.16 (Diagramme) Un diagramme sur une catégorie $\mathcal{C}$ est un couple

$$\overline{\alpha} = (\alpha^\Phi,\ \alpha : \alpha^\Phi \to \mathcal{C}),$$

avec :

- α^Φ est *une catégorie librement engendrée* sur un graphe (ce graphe est également noté α^Φ) ;
- $\alpha : \alpha^\Phi \to \mathcal{C}$ est un foncteur.

Le graphe α^Φ est appelé *graphe sous-jacent* du diagramme $\overline{\alpha}$.

Un diagramme $\overline{\alpha}$ est fini si et seulement si son graphe sous-jacent α^Φ est fini.

Remarque 2.3

1. S. Mac Lane [McL71] définit un diagramme comme un foncteur $F : J \to \mathcal{C}$, où J est une catégorie quelconque ("habituellement petite et souvent finie"). Notre définition est donc moins générale, dans la mesure où nous imposons que la catégorie α^Φ soit librement engendrée sur un graphe, donc petite. De plus la catégorie J peut contenir des égalités entre certaines flèches, ce qui n'est pas le cas pour une catégorie librement engendrée sur un graphe.

2. M. Barr et C. Wells [BW90] définissent un diagramme comme un morphisme de graphes entre le graphe α^Φ et le graphe sous-jacent de la catégorie $\mathcal{C}$. Leur définition est donc un peu moins générale que celle que nous utilisons, puisqu'ils imposent que la catégorie $\mathcal{C}$ soit petite.

3. Lorsque le graphe α^Φ est fini, cela n'implique pas forcément que la catégorie α^Φ soit finie. En effet, si le graphe comporte une ou plusieurs boucles, c'est-à-dire des arcs qui ont mêmes source et but, alors la catégorie librement engendrée sur α^Φ comporte un nombre infini de flèches. La définition de diagramme fini donnée par S. Mac Lane, qui consiste à considérer un foncteur $\alpha : \alpha^\Phi \to \mathcal{C}$ où la catégorie α^Φ est finie, exclut donc les diagrammes dont le graphe sous-jacent comporte des boucles.

4. Nous définissons un diagramme à partir d'un graphe parce que nous pensons que cette définition est plus proche de l'informatique : on a au départ, de façon effective, un graphe, et le fait de considérer la catégorie engendrée par ce graphe doit être vu comme un simple outil de formalisation.

5. Pour définir un foncteur $\alpha : \alpha^\Phi \to \mathcal{C}$ où α^Φ est une catégorie librement engendrée sur un graphe, il suffit de définir α sur les nœuds et les arcs du *graphe* α^Φ. Comme les flèches de la catégorie α^Φ sont les chaînes d'arcs composables sur le graphe α^Φ, il suffit de poser

$$\begin{aligned} &\alpha(\varepsilon_n) = id_{\alpha(n)} \\ &\alpha(a_1 a_2 \dots a_n) = \alpha(a_n) \circ \dots \circ \alpha(a_2) \circ \alpha(a_1) \end{aligned}$$

pour terminer la définition du foncteur $\alpha : \alpha^\Phi \to \mathcal{C}$.

6. Lorsque nous définissons un diagramme $\overline{\alpha}$, nous ne tenons pas compte des noms des nœuds et des arcs du graphe sous-jacent α^Φ. Autrement dit, le graphe α^Φ est considéré *à un isomorphisme près.* Plus précisément, soit deux diagrammes

$$\overline{\alpha} = (\alpha^\Phi,\ \alpha : \alpha^\Phi \to \mathcal{C}) \quad \text{et} \quad \overline{\beta} = (\beta^\Phi,\ \beta : \beta^\Phi \to \mathcal{C}).$$

Si on a un isomorphisme de graphes $\sigma^\Phi : \alpha^\Phi \to \beta^\Phi$ (qui correspond à un foncteur entre les catégories α^Φ et β^Φ) tel que

$$\begin{aligned} &\alpha = \beta \circ \sigma^\Phi \\ &\quad \Leftrightarrow \begin{cases} \forall n \in \text{Nœud}(\alpha^\Phi),\ \alpha(n) = \beta(\sigma^\Phi(n)) \\ \forall a : n \to n' \in \text{Arc}(\alpha^\Phi),\ \alpha(a) = \beta(\sigma^\Phi(a)) \end{cases} \end{aligned}$$

alors, nous considérons que les diagrammes $\overline{\alpha}$ et $\overline{\beta}$ sont égaux.

Nous donnons maintenant quelques exemples de diagrammes sur une catégorie $\mathcal{C}$.

Exemple 2.2 (Diagramme $I_{\mathcal{C}}(A)$) Soit 1^{Φ} le graphe comportant un seul nœud, appelé $*$, et aucun arc. Soit A un objet de $\mathcal{C}$. On a un foncteur $I_{\mathcal{C}}^{A} : 1^{\Phi} \to \mathcal{C}$ défini par $I_{\mathcal{C}}^{A}(*) = A$. On peut donc définir un diagramme

$$I_{\mathcal{C}}(A) = (1^{\Phi},\ I_{\mathcal{C}}^{A} : 1^{\Phi} \to \mathcal{C}).$$



graphe 1^{Φ} diagramme $I_{\mathcal{C}}(A)$

Exemple 2.3 (Diagramme de somme) Soit 2^{Φ} le graphe comportant deux nœuds, et aucun arc. Soit A et B deux objets de $\mathcal{C}$. On a un foncteur $\alpha : 2^{\Phi} \to \mathcal{C}$, qui associe l'un des nœuds à A et l'autre des nœuds à B. On a donc un *diagramme de somme de A et B*

$$\overline{\alpha} = (2^{\Phi},\ \alpha : 2^{\Phi} \to \mathcal{C}).$$



graphe 2^{Φ} diagramme $\overline{\alpha}$

Exemple 2.4 (Diagramme de coégalisateur) Soit κ^{Φ} le graphe contenant deux nœuds 0 et 1, et deux arcs a et b de source 0 et de but 1. Si A et B sont deux objets de $\mathcal{C}$, et $f, g : A \to B$ deux flèches de $\mathcal{C}$, on a *diagramme de coégalisateur*

$$\overline{\kappa} = (\kappa^{\Phi},\ \kappa : \kappa^{\Phi} \to \mathcal{C})$$

défini par

$$\kappa(0) = A$$
$$\kappa(1) = B$$
$$\kappa(a) = f$$
$$\kappa(b) = g.$$



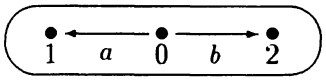
graphe κ^{Φ} diagramme $\overline{\kappa}$

Exemple 2.5 (Diagramme de somme amalgamée) Considérons le graphe π^Φ, appelé "graphe de somme amalgamée", qui comporte trois nœuds 0, 1 et 2, et deux arcs $a : 0 \to 1$ et $b : 0 \to 2$. Si A, B, C sont des objets de $\mathcal{C}$ et $f : A \to B$, $g : A \to C$ des flèches de $\mathcal{C}$, on a un *diagramme de somme amalgamée*

$$\overline{\pi} = (\pi^\Phi,\ \pi : \pi^\Phi \to \mathcal{C})$$

défini par

$$\begin{aligned} \pi(0) &= A \\ \pi(1) &= B \\ \pi(2) &= C \\ \pi(a) &= f \\ \pi(b) &= g. \end{aligned}$$



graphe π^Φ

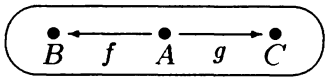


diagramme $\overline{\pi}$

Morphismes de diagrammes

Nous venons de définir les diagrammes finis. Pour obtenir une structure de catégorie, nous devons ajouter des flèches entre les diagrammes. Intuitivement, une flèche entre deux diagrammes $\overline{\alpha}$ et $\overline{\beta}$ sur $\mathcal{C}$ est un ensemble de flèches de $\mathcal{C}$, une par nœud de α^Φ, qui ont certaines propriétés de commutation.

Différentes catégories de diagrammes ont été présentées dans la littérature. Par exemple, S. Mac Lane définit une catégorie de diagrammes, qu'il appelle "super-comma catégorie" ([McL71], exercice 5.(b), page 111). A. Tarlecki *et al.* ont défini une catégorie **Funct**($\mathcal{C}$) [TBG91], qui peut être obtenue à partir de la "super-comma catégorie" de S. Mac Lane en dualisant. Dans cette catégorie, une flèche entre deux diagrammes

$$\overline{\alpha} = (\alpha^\Phi,\ \alpha : \alpha^\Phi \to \mathcal{C}) \quad \text{et} \quad \overline{\beta} = (\beta^\Phi,\ \beta : \beta^\Phi \to \mathcal{C})$$

est un couple

$$\overline{\sigma} : \overline{\alpha} \to \overline{\beta} = (\sigma^\Phi : \alpha^\Phi \to \beta^\Phi\ ;\ \sigma : \alpha \dot{\to} \beta \circ \sigma^\Phi)$$

où

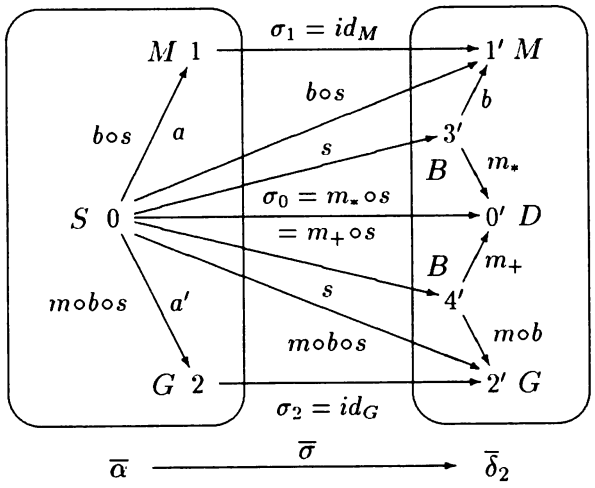
- σ^Φ est un foncteur entre les catégories α^Φ et β^Φ, ou, de façon équivalente, un morphisme de graphes entre les graphes α^Φ et β^Φ ;
- $\sigma : \alpha \dot{\to} \beta \circ \sigma^\Phi$ est une transformation naturelle entre les foncteurs $\alpha : \alpha^\Phi \to \mathcal{C}$ et $\beta \circ \sigma^\Phi : \alpha^\Phi \to \mathcal{C}$.

Ces catégories de diagrammes ne sont pas suffisamment générales pour notre propos, car elles ne comportent pas assez de flèches. Intuitivement, et en anticipant un peu sur la suite, nous souhaitons avoir une flèche entre deux diagrammes chaque fois qu'il existe une flèche entre les *colimites* des deux diagrammes. Avec la définition ci-dessus, certaines flèches entre colimites de diagrammes ne correspondent à aucune flèche entre les deux diagrammes.

Exemple 2.6 Reprenons l'exemple des anneaux du chapitre 1. Nous avons un morphisme de diagrammes du diagramme $\overline{\alpha}$ vers le diagramme $\overline{\delta_2}$ qui correspond à un morphisme de spécifications de Colim $\overline{\alpha}$ vers Colim $\overline{\delta_2}$:

$$\mathsf{up}\,(S, M, G, A_2, b \circ s, m \circ b \circ s, \&_1(A_2) \circ \&_1(MD), \&_2(A_2)).$$

Mais il n'y a pas de morphisme de diagrammes de $\overline{\alpha}$ vers $\overline{\delta_2}$ avec la définition ci-dessus.



Pour cette raison, au lieu de considérer un morphisme de graphes $\sigma^\Phi : \alpha^\Phi \to \beta^\Phi$ nous devons considérer un *morphisme généralisé* de graphes $\sigma^\Phi : \alpha^\Phi \rightsquigarrow \beta^\Phi$. Rappelons qu'un morphisme généralisé de graphes associe à tout nœud de α^Φ un nœud de β^Φ, et à tout arc de α^Φ un *zigzag* de β^Φ, c'est-à-dire une séquence linéaire d'arcs de β^Φ (définition 2.5). De même, au lieu de considérer une transformation naturelle $\sigma : \alpha \dot{\to} \beta \circ \sigma^\Phi$, nous devons considérer une *transformation naturelle généralisée* $\sigma : \alpha \dot{\rightsquigarrow} \beta \circ \sigma^\Phi$.

Exemple 2.6 (Suite) Avec cette définition, nous pouvons définir une flèche $\overline{\sigma} : \overline{\alpha} \to \overline{\delta_2}$, qui consiste en un morphisme généralisé de graphes $\sigma^\Phi : \alpha^\Phi \rightsquigarrow \delta_2^\Phi$ et une transformation naturelle généralisée $\sigma : \alpha \dot{\rightsquigarrow} \delta_2 \circ \sigma^\Phi$.

- Le morphisme généralisé de graphes σ^Φ peut être défini par exemple de la

façon suivante :

$$\begin{array}{rcl} \sigma^\Phi : \alpha^\Phi & \rightsquigarrow & \delta_2^\Phi \\ 0 & \mapsto & 0' \\ 1 & \mapsto & 1' \\ 2 & \mapsto & 2' \\ a & \mapsto & 0' \xleftarrow{m_*} 3' \xrightarrow{b} 1' \\ a' & \mapsto & 0' \xleftarrow{m_+} 4' \xrightarrow{mob} 2' \end{array}$$

- La transformation naturelle généralisée $\sigma : \alpha \rightsquigarrow \delta_2 \circ \sigma^\Phi$ est définie par :

$$\sigma_0 = m_* \circ s = m_+ \circ s$$
$$\sigma_1 = id_M$$
$$\sigma_2 = id_G.$$

Définition 2.17 (Relation de connexion entre deux flèches)
Soit un diagramme $\overline{\delta} = (\delta^\Phi,\ \delta : \delta^\Phi \to \mathcal{C})$, et deux nœuds n_0 et n_k de Nœud(δ^Φ).
Soit $u : A \to \delta(n_0)$ et $v : A \to \delta(n_k)$ deux flèches de $\mathcal{C}$.
On dit que les flèches u et v sont *connectées par le diagramme* $\overline{\delta}$ (figure 2.1) si et seulement si il existe un zigzag

$$Z \ = \ n_0 \xrightarrow{a_0} n_1 \xleftarrow{a_1} n_2 \xleftarrow{a_2} n_2 \cdots n_{k-1} \xrightarrow{a_{k-1}} n_k$$

sur δ^Φ, et un ensemble de flèches de $\mathcal{C}$

$$\{c_i : A \to \delta(n_i)\ ;\ i \in \{0, \ldots, k\}\},$$

tels que :

- $u = c_0$;
- $v = c_k$:
- $\forall i \in \{0, \ldots, k-1\}$:
  $\delta(a_i) \circ c_i = c_{i+1}$, si a_i est orienté de n_i vers n_{i+1} ;
  $\delta(a_i) \circ c_{i+1} = c_i$, si a_i est orienté de n_{i+1} vers n_i.

On note $u \sim_{\overline{\delta}} v$; ou $u \sim_{\overline{\delta}} v\ [Z]$, si l'on souhaite préciser le zigzag Z.

Définition 2.18 (Catégorie des diagrammes finis DIAGR($\mathcal{C}$))
Soit une catégorie $\mathcal{C}$.

- Un *objet* $\overline{\delta}$ de DIAGR($\mathcal{C}$) est un diagramme fini sur $\mathcal{C}$.
- Soit deux diagrammes $\overline{\alpha} = (\alpha^\Phi,\ \alpha : \alpha^\Phi \to \mathcal{C})$ et $\overline{\beta} = (\beta^\Phi,\ \beta : \beta^\Phi \to \mathcal{C})$. Une *flèche* $\overline{\tau} : \overline{\alpha} \to \overline{\beta}$ de DIAGR($\mathcal{C}$), ou *morphisme de diagrammes*, est un couple

$$\overline{\tau} : \overline{\alpha} \to \overline{\beta} = (\tau^\Phi : \alpha^\Phi \rightsquigarrow \beta^\Phi,\ \tau : \alpha \rightsquigarrow \beta \circ \tau^\Phi),$$

avec :

  - $\tau^\Phi : \alpha^\Phi \rightsquigarrow \beta^\Phi$ est un morphisme généralisé de graphes.

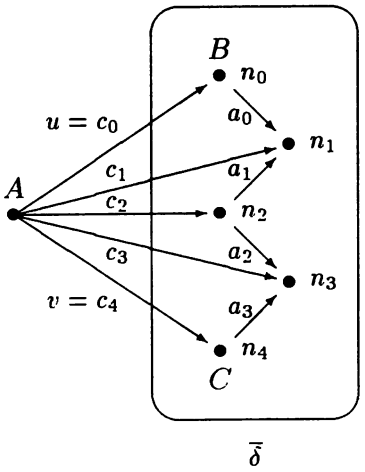


FIG. 2.1 – : $u \sim_{\overline{\delta}} v$ $[Z : n_0 \rightsquigarrow n_4]$

– $\tau : \alpha \rightsquigarrow \beta \circ \tau^\Phi$ est une "transformation naturelle généralisée", c'est-à-dire un ensemble de flèches

$$\tau_n : \alpha(n) \to \beta(\tau^\Phi(n)),\ \forall n \in \text{Nœud}(\alpha^\Phi)$$

telles que

$$\forall a : m \to n \in \text{Arc}(\alpha^\Phi),\ \ \tau_n \circ \alpha(a) \sim_{\overline{\beta}} \tau_m\ \ [\tau^\Phi(a)].$$

Remarquons que dans le cas particulier où τ^Φ est un morphisme de graphes, alors τ^Φ s'étend en un foncteur $\tau^\Phi : \alpha^\Phi \to \beta^\Phi$, et τ est alors une transformation naturelle $\tau : \alpha \dot{\to} \beta \circ \tau^\Phi$.

• On définit la composition de deux flèches de DIAGR($\mathcal{C}$) de la façon suivante :

Soit trois diagrammes :

$$\overline{\alpha} = (\alpha^\Phi,\ \alpha : \alpha^\Phi \to \mathcal{C});$$
$$\overline{\beta} = (\beta^\Phi,\ \beta : \beta^\Phi \to \mathcal{C});$$
$$\overline{\gamma} = (\gamma^\Phi,\ \gamma : \gamma^\Phi \to \mathcal{C}).$$

Soit deux flèches :

$$\overline{\sigma} : \overline{\alpha} \to \overline{\beta} = (\sigma^\Phi : \alpha^\Phi \rightsquigarrow \beta^\Phi,\ \sigma : \alpha \rightsquigarrow \beta \circ \sigma^\Phi);$$
$$\overline{\tau} : \overline{\beta} \to \overline{\gamma} = (\tau^\Phi : \beta^\Phi \rightsquigarrow \gamma^\Phi,\ \tau : \beta \rightsquigarrow \gamma \circ \tau^\Phi).$$

La composition $\overline{\tau} \circ \overline{\sigma}$ de $\overline{\sigma}$ et $\overline{\tau}$ est le couple

$$\overline{\tau} \circ \overline{\sigma} : \overline{\alpha} \to \overline{\gamma} = (\tau^\Phi \circ \sigma^\Phi : \alpha^\Phi \rightsquigarrow \gamma^\Phi,\ \lambda : \alpha \rightsquigarrow \gamma \circ \tau^\Phi \circ \sigma^\Phi)$$

où

– $\tau^\Phi \circ \sigma^\Phi$ est la composition des deux morphismes de graphes généralisés ;

– $\lambda : \alpha \rightsquigarrow \gamma \circ \tau^\Phi \circ \sigma^\Phi$ est la *transformation naturelle généralisée* définie par

$$\forall n \in \text{Nœud}(\alpha^\Phi),\ \lambda_n = \tau_{\sigma^\Phi(n)} \circ \sigma_n.$$

$$\alpha(n) \xrightarrow{\sigma_n} \beta(\sigma^\Phi(n)) \xrightarrow{\tau_{\sigma^\Phi(n)}} \gamma.(\tau^\Phi(\sigma^\Phi(n)))$$

On vérifie sans difficulté que λ est bien une transformation naturelle généralisée, c'est-à-dire que

$$\forall a : m \to n \in \text{Arc}(\alpha^\Phi),\ \tau_{\sigma^\Phi(n)} \circ \sigma_n \circ \alpha(a) \sim_{\overline{\gamma}} \tau_{\sigma^\Phi(m)} \circ \sigma_m\ \ [\tau^\Phi(\sigma^\Phi(a))].$$

- On vérifie que cette composition est associative.
- Pour tout diagramme $\overline{\alpha}$, on a une flèche identité

$$\overline{id}_{\overline{\alpha}} : \overline{\alpha} \to \overline{\alpha} = (id_{\alpha^\Phi} : \alpha^\Phi \rightsquigarrow \alpha^\Phi,\ id_{\overline{\alpha}} : \alpha \rightsquigarrow \alpha),$$

où $id_{\overline{\alpha}} : \alpha \rightsquigarrow \alpha$ est la transformation naturelle identité $id_\alpha : \alpha \dot{\to} \alpha$. Autrement dit, $\forall n \in \text{Nœud}(\alpha^\Phi)$, $(id_{\overline{\alpha}})_n = id_{\alpha(n)}$.

2.3.2 Colimites

La façon standard de définir une colimite (comme par exemple dans [McL71]) est de commencer par fixer un graphe sous-jacent α^Φ, puis de définir un foncteur diagonal

$$\Delta : \mathcal{C} \to \mathcal{C}^{\alpha^\Phi},$$

où $\mathcal{C}^{\alpha^\Phi}$ est la catégorie des foncteurs de α^Φ vers $\mathcal{C}$, qui correspond à une catégorie de diagrammes de graphe sous-jacent α^Φ. Cette présentation ne nous convient pas, parce que nous ne voulons pas fixer de graphe *a priori*. En définissant la catégorie DIAGR($\mathcal{C}$), nous n'avons pas non plus fixé de graphe sous-jacent, puisqu'un diagramme est donné à la fois par un *graphe* et un *foncteur*.

Dans ce paragraphe, nous reformulons la définition de colimite dans notre cadre. Cette définition correspond à la définition standard, et évite de fixer trop tôt un graphe sous-jacent.

Définition 2.19 (Foncteur d'inclusion $I_\mathcal{C} : \mathcal{C} \to \text{DIAGR}(\mathcal{C})$)
La catégorie $\mathcal{C}$ peut être plongée dans DIAGR($\mathcal{C}$) en considérant tout objet de $\mathcal{C}$ comme un diagramme sur le graphe 1^Φ. Autrement dit, on a un foncteur

$$\begin{array}{rccc} I_\mathcal{C} : & \mathcal{C} & \to & \text{DIAGR}(\mathcal{C}) \\ & A & \mapsto & \bullet\ A \\ & f : A \to B & \mapsto & A\ \bullet \xrightarrow{f} \bullet\ B \end{array}$$

Le foncteur correspondant au diagramme $I_{\mathcal{C}}(A)$ est noté $I_{\mathcal{C}}^{A} : 1^{\Phi} \to \mathcal{C}$.
La transformation naturelle généralisée correspondant au morphisme de diagrammes $I_{\mathcal{C}}(f)$ est notée $I_{\mathcal{C}}^{f} : I_{\mathcal{C}}^{A} \rightsquigarrow I_{\mathcal{C}}^{B}$.
Nous avons donc :

$$I_{\mathcal{C}}(A) = (1^{\Phi},\ I_{\mathcal{C}}^{A} : 1^{\Phi} \to \mathcal{C})$$
$$I_{\mathcal{C}}(B) = (1^{\Phi},\ I_{\mathcal{C}}^{B} : 1^{\Phi} \to \mathcal{C})$$
$$I_{\mathcal{C}}(f) = (id_{1^{\Phi}},\ I_{\mathcal{C}}^{f} : I_{\mathcal{C}}^{A} \rightsquigarrow I_{\mathcal{C}}^{B}).$$

Soit $*$ l'unique nœud de 1^{Φ}. Nous avons :

$$I_{\mathcal{C}}^{A}(*) = A$$
$$I_{\mathcal{C}}^{B}(*) = B$$
$$(I_{\mathcal{C}}^{f})_{*} = f : A \to B.$$

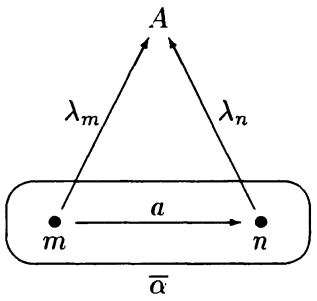
Définition 2.20 (Cône) Soit un diagramme $\overline{\alpha} = (\alpha^{\Phi},\ \alpha : \alpha^{\Phi} \to \mathcal{C})$. Un *cône*[1] sur $\overline{\alpha}$ est un couple

$$(A,\ \overline{\lambda} : \overline{\alpha} \to I_{\mathcal{C}}(A))$$

où A est un objet de $\mathcal{C}$, et $\overline{\lambda}$ est une flèche entre les diagrammes $\overline{\alpha}$ et $I_{\mathcal{C}}(A)$.
λ^{Φ} est l'unique morphisme généralisé de α^{Φ} vers 1^{Φ} : à tout nœud de α^{Φ}, λ^{Φ} associe l'unique nœud $*$ de 1^{Φ}, et à tout arc de α^{Φ}, λ^{Φ} associe le zigzag de longueur nulle $0_{*} : * \rightsquigarrow *$. Remarquons que $0_{*} : * \rightsquigarrow *$ est l'unique zigzag de 1^{Φ}.

Remarque 2.4 La transformation naturelle généralisée $\lambda : \alpha \rightsquigarrow I_{\mathcal{C}}^{A} \circ \lambda^{\Phi}$ est en fait une transformation naturelle $\lambda : \alpha \dot{\to} I_{\mathcal{C}}^{A} \circ \lambda^{\Phi}$. En détaillant, on a

$$\forall m, n \in \text{Nœud}(\alpha^{\Phi}),\ \forall a : m \to n \in \text{Arc}(\alpha^{\Phi}),\ \lambda_{n} \circ \alpha(a) = \lambda_{m}.$$



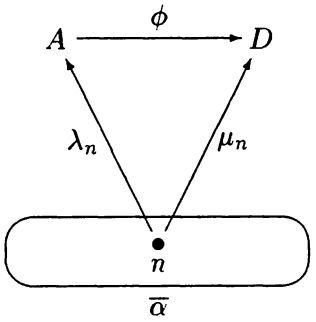
Définition 2.21 (Cône colimite)
Le cône $(A,\ \overline{\lambda} : \overline{\alpha} \to I_{\mathcal{C}}(A))$ est un *cône colimite* du diagramme $\overline{\alpha}$ si et seulement si pour tout cône $(D,\ \overline{\mu} : \overline{\alpha} \to I_{\mathcal{C}}(D))$, il existe une unique flèche $\phi : A \to D$ de $\mathcal{C}$ telle que

$$I_{\mathcal{C}}(\phi) \circ \overline{\lambda} = \overline{\mu}.$$

1. Pour respecter la terminologie usuelle, nous devrions parler de *cône inductif* ou de *cocône*, car nous allons définir des *colimites* et non des *limites*. Dans la mesure où nous ne parlons ici jamais de limites, nous utiliserons le mot *cône*.

Détaillons un peu cette dernière égalité.

$$\begin{aligned} & I_{\mathcal{C}}(\phi) \circ \overline{\lambda} = \overline{\mu} \\ & \quad \Leftrightarrow \quad \forall n \in \text{Nœud}(\alpha^{\Phi}),\ I_{\mathcal{C}}(\phi)_{\lambda^{\Phi}(n)} \circ \lambda_n = \mu_n \\ & \quad \Leftrightarrow \quad \forall n \in \text{Nœud}(\alpha^{\Phi}),\ \phi \circ \lambda_n = \mu_n \end{aligned}$$



Si $(A,\ \overline{\lambda} : \overline{\alpha} \to I_{\mathcal{C}}(A))$ est un cône colimite, on appelle A la *colimite* du diagramme $\overline{\alpha}$, et on note $A = \text{Colim}_{\mathcal{C}}\, \overline{\alpha}$. Cette notation compacte ne doit pas faire oublier qu'en réalité, une colimite A est toujours définie en même temps qu'un cône, c'est-à-dire une flèche $\overline{\lambda} : \overline{\alpha} \to I_{\mathcal{C}}(A)$.

Les deux lemmes suivants montrent qu'une colimite est définie à un isomorphisme près.

Lemme 2.1 *(Deux colimites d'un même diagramme sont isomorphes)*
Soit un diagramme $\overline{\alpha}$ sur $\mathcal{C}$. Si $(A,\ \overline{\lambda} : \overline{\alpha} \to I_{\mathcal{C}}(A))$ et $(B,\ \overline{\mu} : \overline{\alpha} \to I_{\mathcal{C}}(B))$ sont deux cônes colimites du diagramme $\overline{\alpha}$, alors $A \cong B$.

Preuve. Comme $(B,\ \overline{\mu} : \overline{\alpha} \to I_{\mathcal{C}}(B))$ est un cône sur $\overline{\alpha}$, il existe une unique flèche $\phi : A \to B$ telle que

$$I_{\mathcal{C}}(\phi) \circ \overline{\lambda} = \overline{\mu} \qquad (i)$$

Comme $(A,\ \overline{\lambda} : \overline{\alpha} \to I_{\mathcal{C}}(A))$ est un cône sur $\overline{\alpha}$, il existe une unique flèche $\psi : B \to A$ telle que

$$I_{\mathcal{C}}(\psi) \circ \overline{\mu} = \overline{\lambda} \qquad (ii)$$

Nous avons donc

$$\begin{aligned} I_{\mathcal{C}}(\psi \circ \phi) \circ \overline{\lambda} &= I_{\mathcal{C}}(\psi) \circ I_{\mathcal{C}}(\phi) \circ \overline{\lambda} && (I_{\mathcal{C}} \text{ foncteur}) \\ &= I_{\mathcal{C}}(\psi) \circ \overline{\mu} && (\text{d'après } (i)) \\ &= \overline{\lambda} && (\text{d'après } (ii)) \end{aligned}$$

Comme $(A,\ \overline{\lambda} : \overline{\alpha} \to I_{\mathcal{C}}(A))$ est un cône colimite, il existe une unique flèche $u : A \to A$ telle que $I_{\mathcal{C}}(u) \circ \overline{\lambda} = \overline{\lambda}$. Or $\psi \circ \phi$ et id_A satisfont toutes les deux cette propriété, donc $\psi \circ \phi = id_A$. De même, on montre que $\phi \circ \psi = id_B$. Par conséquent, A et B sont isomorphes. □

Lemme 2.2 *Soit un diagramme $\overline{\alpha}$ sur $\mathcal{C}$, ayant pour cône colimite*

$$(A,\ \overline{\lambda} : \overline{\alpha} \to I_{\mathcal{C}}(A)).$$

Soit B un objet de $\mathcal{C}$ isomorphe à A. On note $\phi : A \to B$ cet isomorphisme. Alors $(B,\ I_{\mathcal{C}}(\phi) \circ \overline{\lambda} : \overline{\alpha} \to I_{\mathcal{C}}(B))$ est un (autre) cône colimite du diagramme $\overline{\alpha}$.

Preuve. Il faut montrer que pour tout cône $(D,\ \overline{\mu} : \overline{\alpha} \to I_{\mathcal{C}}(D))$, il existe un unique $\psi : B \to D$ tel que $I_{\mathcal{C}}(\psi) \circ I_{\mathcal{C}}(\phi) \circ \overline{\lambda} = \overline{\mu}$.

Existence. Soit $\xi : A \to D$ l'unique flèche telle que $I_{\mathcal{C}}(\xi) \circ \overline{\lambda} = \overline{\mu}$. On pose

$$\psi = \xi \circ \phi^{-1}.$$

$$\begin{array}{lll} I_{\mathcal{C}}(\psi) \circ I_{\mathcal{C}}(\phi) \circ \overline{\lambda} & & \\ \quad = I_{\mathcal{C}}(\xi \circ \phi^{-1}) \circ I_{\mathcal{C}}(\phi) \circ \overline{\lambda} & \text{(définition de } \psi) \\ \quad = I_{\mathcal{C}}(\xi) \circ \overline{\lambda} & (I_{\mathcal{C}} \text{ foncteur}) \\ \quad = \overline{\mu} & \text{(définition de } \xi) \end{array}$$

Unicité. Soit $\psi' : B \to D$ tel que $I_{\mathcal{C}}(\psi') \circ I_{\mathcal{C}}(\phi) \circ \overline{\lambda} = \overline{\mu}$. On montre que $\psi = \psi'$.

$$\begin{array}{ll} I_{\mathcal{C}}(\psi) \circ I_{\mathcal{C}}(\phi) \circ \overline{\lambda} = I_{\mathcal{C}}(\psi') \circ I_{\mathcal{C}}(\phi) \circ \overline{\lambda} = \overline{\mu} & \text{(définitions de } \psi \text{ et } \psi') \\ \quad \Rightarrow\ I_{\mathcal{C}}(\psi \circ \phi) \circ \overline{\lambda} = I_{\mathcal{C}}(\psi' \circ \phi) \circ \overline{\lambda} = \overline{\mu} & (I_{\mathcal{C}} \text{ foncteur}) \end{array}$$

Comme $(A,\ \overline{\lambda} : \overline{\alpha} \to I_{\mathcal{C}}(A))$ est cône colimite du diagramme $\overline{\alpha}$, il existe une unique flèche $u : A \to D$ de $\mathcal{C}$ telle que

$$I_{\mathcal{C}}(u) \circ \overline{\lambda} = \overline{\mu}.$$

Comme $\psi \circ \phi$ et $\psi' \circ \phi$ satisfont cette propriété,

$$\psi \circ \phi = \psi' \circ \phi.$$

Or ϕ est un isomorphisme, donc $\psi = \psi'$.

□

Remarque 2.5 (Choix de colimites)

Une colimite (avec le cône colimite correspondant) est donc définie à un isomorphisme près. On parle souvent abusivement de *la* colimite d'un diagramme, alors qu'on devrait parler d'*une* colimite d'un diagramme. Pour la suite, nous avons besoin de distinguer une colimite particulière pour un diagramme. Cela consiste à *choisir* un cône colimite privilégié pour ce diagramme. Dans ce cas, nous pouvons alors parler sans ambiguïté de *la* colimite de ce diagramme.

Lorsqu'on n'a pas fait de choix de colimites, la notation $\mathrm{Colim}_{\mathcal{C}}\ \overline{\alpha}$ représente donc *une colimite quelconque* du diagramme $\overline{\alpha}$ (sachant qu'elles sont toutes isomorphes). Par contre, lorsqu'on a fait un choix de colimite, la notation $\mathrm{Colim}_{\mathcal{C}}\ \overline{\alpha}$ représente *la colimite choisie* du diagramme $\overline{\alpha}$.

Ces choix de colimites correspondent à des choix de représentation ou de codage, nécessaires en informatique, en particulier lorsqu'on définit une syntaxe.

2.3.3 Exemples de colimites

Dans ce paragraphe, nous considérons quelques exemples de colimites de diagrammes particuliers sur une catégorie $\mathcal{C}$. Nous rappelons d'abord la définition d'objet initial.

Définition 2.22 (Objet initial) Un objet $\emptyset$ de $\mathcal{C}$ est initial si et seulement si pour tout objet A de $\mathcal{C}$, il existe une unique flèche $j_A : \emptyset \to A$ dans $\mathcal{C}$.

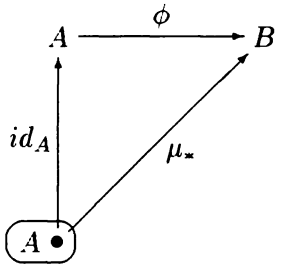
Exemple 2.7 (Objet initial)
L'ensemble vide est initial dans la catégorie **Set**.
La spécification vide est initiale dans la catégorie **Spec**.
Le diagramme vide $\bigcirc$ (diagramme dont le graphe sous-jacent ne comporte aucun nœud et aucun arc) est initial dans DIAGR($\mathcal{C}$).

Si le diagramme vide $\bigcirc$ a une colimite $\emptyset$ dans $\mathcal{C}$, alors $\emptyset$ est un objet initial de $\mathcal{C}$. Soit $(\emptyset, \overline{\lambda} : \bigcirc \to I_{\mathcal{C}}(\emptyset))$ le cône colimite de $\bigcirc$. Soit A un objet de $\mathcal{C}$. Comme $\bigcirc$ est initial dans DIAGR($\mathcal{C}$), on a une flèche $\overline{\mu} : \bigcirc \to I_{\mathcal{C}}(A)$. $\emptyset$ est colimite du diagramme $\bigcirc$, donc il existe une unique flèche $j_A : \emptyset \to A$ telle que $I_{\mathcal{C}}(j_A) \circ \overline{\lambda} = \overline{\mu}$. Soit $j : \emptyset \to A$ une autre flèche de $\mathcal{C}$. On a $I_{\mathcal{C}}(j) \circ \overline{\lambda} = \overline{\mu}$ car $\bigcirc$ est initial dans DIAGR($\mathcal{C}$). Par conséquent, $j = j_A$.

Exemple 2.8 (Colimite du diagramme $I_{\mathcal{C}}(A)$) Soit un objet A de $\mathcal{C}$. Considérons le diagramme $\overline{\alpha} = I_{\mathcal{C}}(A)$. Le cône $(A, \overline{id}_{\overline{\alpha}} : \overline{\alpha} \to I_{\mathcal{C}}(A))$ est un cône colimite de $\overline{\alpha}$.

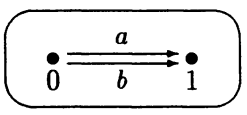
Preuve. Soit $(B, \overline{\mu} : \overline{\alpha} \to I_{\mathcal{C}}(B))$ un cône sur $\overline{\alpha}$. Il existe une unique flèche $\phi : A \to B$ de $\mathcal{C}$ telle que $I_{\mathcal{C}}(\phi) \circ \overline{id}_{\overline{\alpha}} = \overline{\mu}$. En effet,

$$\begin{aligned} & I_{\mathcal{C}}(\phi) \circ \overline{id}_{\overline{\alpha}} = \overline{\mu} \\ \Leftrightarrow \quad & \phi \circ id_A = \mu_* \\ \Leftrightarrow \quad & \phi = \mu_* \end{aligned}$$



□

Exemple 2.9 (Coégalisateur) Soit deux objets A et B, ainsi que deux flèches $f, g : A \to B$. Le coégalisateur de f et g est la colimite du diagramme $\overline{\kappa}$, construit sur le graphe κ^{Φ}.



graphe κ^{Φ}

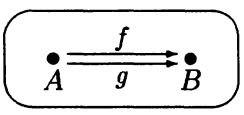


diagramme $\overline{\kappa}$

Soit un cône $(D, \overline{\mu} : \overline{\kappa} \to I_{\mathcal{C}}(D))$ sur $\overline{\kappa}$. Ce cône est entièrement déterminé par une flèche $h : B \to D$ telle que

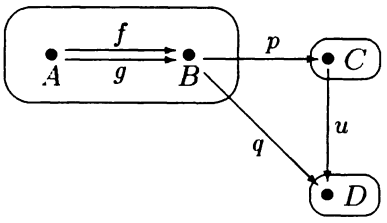
$$h \circ f = h \circ g.$$

En effet, il suffit de poser

$$\mu_0 = h \circ f = h \circ g$$
$$\mu_1 = h$$

pour définir la flèche $\overline{\mu} : \overline{\kappa} \to I_{\mathcal{C}}(D)$.

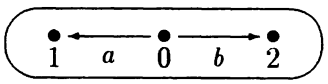
Soit $(C, \overline{\lambda} : \overline{\kappa} \to I_{\mathcal{C}}(C))$ le cône colimite de $\overline{\kappa}$. Alors, pour tout objet D et flèche $q : B \to D$ telle que $q \circ f = q \circ g$, il existe une unique flèche $u : C \to D$ telle que $u \circ p = q$.



Dans la catégorie des ensembles **Set**, le coégalisateur C est le quotient de l'ensemble B par la relation d'équivalence engendrée par $\{(f(a), g(a)), \forall a \in A\}$. L'application $p : B \to C$ est la projection de B sur le quotient C. Une application $q : B \to D$ telle que $q \circ f = q \circ g$ est une application compatible avec la relation d'équivalence, qui se factorise donc à travers p. Il existe par conséquent une unique application $u : C' \to D$ telle que $u \circ p = q$.

La construction de somme amalgamée de deux ensembles proposée dans le chapitre 1 (section 1.4.1) est en réalité une façon générale de construire une somme amalgamée à partir d'une somme et d'un coégalisateur.

Exemple 2.10 (Somme amalgamée) Soit trois objets A, B, C et deux flèches $f : A \to B$, $g : A \to C$ dans $\mathcal{C}$. La somme amalgamée de B et C par rapport à $f : A \to B$ et $g : A \to C$ est la colimite du diagramme $\overline{\pi}$, construit sur le graphe π^{Φ}.



graphe π^{Φ}

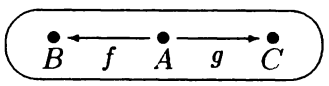


diagramme $\overline{\pi}$

Soit $(P, \overline{\&} : \overline{\pi} \to I_{\mathcal{C}}(P))$ le cône colimite de $\overline{\pi}$. Ce cône colimite consiste en trois flèches $\&_0 : A \to P$, $\&_1 : B \to P$ et $\&_2 : C \to P$ telles que

$$\&_0 = \&_1 \circ f = \&_2 \circ g.$$

Ce cône est donc entièrement déterminé par les deux flèches $\&_1$ et $\&_2$.

Soit $(D,\ \overline{\mu} : \overline{\pi} \to I_C(D))$ un autre cône sur $\overline{\pi}$. Ce cône est constitué de trois flèches $\mu_0 : A \to D$, $f' = \mu_1 : B \to D$ et $g' = \mu_2 : C \to D$ telles que

$$\mu_0 = f' \circ f = g' \circ g.$$

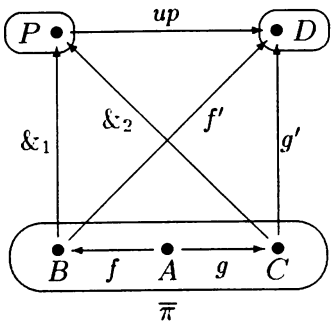
Un cône sur $\overline{\pi}$ est donc entièrement déterminé par deux flèches $f' : B \to D$ et $g' : C \to D$ telles que $f' \circ f = g' \circ g$.
Comme P est la colimite de $\overline{\pi}$, il existe une unique flèche $up : P \to D$ telle que

$$I_C(up) \circ \overline{\&} = \overline{\mu}.$$

Cette égalité est équivalente aux deux égalités

$$up \circ \&_1 = f' \qquad (i)$$
$$up \circ \&_2 = g' \qquad (ii)$$

car l'égalité (i) — comme l'égalité (ii) — implique $up \circ \&_0 = \mu_0$.



somme amalgamée, colimite du diagramme $\overline{\pi}$

Définition 2.23 (Catégorie finiment cocomplète) Une catégorie C est *finiment cocomplète* si et seulement si tout diagramme fini a une colimite.

Théorème 2.2 *Une catégorie C est finiment cocomplète si et seulement si C a un objet initial et des sommes amalgamées.*

Preuve. Pour montrer ce théorème, on utilise le théorème suivant :

Théorème 2.3 *Une catégorie C est finiment cocomplète si et seulement si C a un objet initial, des sommes pour tout couple d'objets de C, et des coégalisateurs pour tout couple de flèches $f, g : A \to B$ de C.*

Le dual de ce théorème est démontré dans [McL71], page 109. On vérifie facilement qu'on peut adapter la démonstration de S. Mac Lane à notre définition de diagramme. Ensuite, pour montrer le théorème 2.2, il suffit de montrer qu'on peut simuler la construction d'une somme et d'un coégalisateur à l'aide d'un objet initial et de sommes amalgamées. □

2.3.4 Adjonction ($\mathrm{Colim}_{\mathcal{C}} \dashv I_{\mathcal{C}}$)

Dans ce paragraphe, nous considérons une catégorie $\mathcal{C}$ finiment cocomplète. Tout diagramme fini

$$\overline{\alpha} = (\alpha^{\Phi},\ \alpha : \alpha^{\Phi} \to \mathcal{C})$$

a donc un cône colimite que l'on va noter

$$(\mathrm{Colim}_{\mathcal{C}}\, \overline{\alpha},\ \overline{\eta}_{\overline{\alpha}} : \overline{\alpha} \to I_{\mathcal{C}}(\mathrm{Colim}_{\mathcal{C}}\, \overline{\alpha})).$$

Nous montrons que l'on peut étendre la fonction $\mathrm{Colim}_{\mathcal{C}}$ sur les flèches de $\mathrm{DIAGR}(\mathcal{C})$ de façon à ce que $\mathrm{Colim}_{\mathcal{C}} : \mathrm{DIAGR}(\mathcal{C}) \to \mathcal{C}$ soit un foncteur. La définition des *flèches* de $\mathrm{DIAGR}(\mathcal{C})$ a été justement choisie afin que cette extension soit possible.

Soit deux diagrammes

$$\overline{\alpha} = (\alpha^{\Phi},\ \alpha : \alpha^{\Phi} \to \mathcal{C}) \quad \text{et} \quad \overline{\beta} = (\beta^{\Phi},\ \beta : \beta^{\Phi} \to \mathcal{C})$$

ayant respectivement pour cônes colimites

$$(\mathrm{Colim}_{\mathcal{C}}\, \overline{\alpha},\ \overline{\eta}_{\overline{\alpha}} : \overline{\alpha} \to I_{\mathcal{C}}(\mathrm{Colim}_{\mathcal{C}}\, \overline{\alpha})) \quad \text{et} \quad (\mathrm{Colim}_{\mathcal{C}}\, \overline{\beta},\ \overline{\eta}_{\overline{\beta}} : \overline{\beta} \to I_{\mathcal{C}}(\mathrm{Colim}_{\mathcal{C}}\, \overline{\beta})).$$

Soit $\overline{\sigma} : \overline{\alpha} \to \overline{\beta} = (\sigma^{\Phi} : \alpha^{\Phi} \rightsquigarrow \beta^{\Phi},\ \sigma : \alpha \rightsquigarrow \beta \circ \sigma^{\Phi})$ une flèche de $\mathrm{DIAGR}(\mathcal{C})$. Considérons le cône sur $\overline{\alpha}$

$$(\mathrm{Colim}_{\mathcal{C}}\, \overline{\beta},\ \overline{\eta}_{\overline{\beta}} \circ \overline{\sigma} : \overline{\alpha} \to I_{\mathcal{C}}(\mathrm{Colim}_{\mathcal{C}}\, \overline{\beta})).$$

Comme $(\mathrm{Colim}_{\mathcal{C}}\, \overline{\alpha},\ \overline{\eta}_{\overline{\alpha}} : \overline{\alpha} \to I_{\mathcal{C}}(\mathrm{Colim}_{\mathcal{C}}\, \overline{\alpha}))$ est un cône colimite de $\overline{\alpha}$, il existe une unique flèche

$$\mathrm{Colim}_{\mathcal{C}}\, \overline{\sigma} : \mathrm{Colim}_{\mathcal{C}}\, \overline{\alpha} \to \mathrm{Colim}_{\mathcal{C}}\, \overline{\beta}$$

telle que

$$I_{\mathcal{C}}(\mathrm{Colim}_{\mathcal{C}}\, \overline{\sigma}) \circ \overline{\eta}_{\overline{\alpha}} = \overline{\eta}_{\overline{\beta}} \circ \overline{\sigma}.$$

Théorème 2.4 *Soit $\mathcal{C}$ une catégorie finiment cocomplète. Alors, on a les résultats suivants.*

1. $\mathrm{Colim}_{\mathcal{C}} : \mathrm{DIAGR}(\mathcal{C}) \to \mathcal{C}$ *est un foncteur.*
2. *La fonction $\overline{\eta}$, qui à tout diagramme $\overline{\alpha}$ associe la flèche du cône colimite de $\overline{\alpha}$ vers $I_{\mathcal{C}}(\mathrm{Colim}_{\mathcal{C}}\, \overline{\alpha})$ est une transformation naturelle*

$$\overline{\eta} : Id_{\mathrm{DIAGR}(\mathcal{C})} \dot{\to} I_{\mathcal{C}} \circ \mathrm{Colim}_{\mathcal{C}}.$$

3. $(\mathrm{Colim}_{\mathcal{C}} \dashv I_{\mathcal{C}}) : \mathrm{DIAGR}(\mathcal{C}) \to \mathcal{C}$ *est une adjonction, dont l'unité est la transformation naturelle*

$$\overline{\eta} : Id_{\mathrm{DIAGR}(\mathcal{C})} \dot{\to} I_{\mathcal{C}} \circ \mathrm{Colim}_{\mathcal{C}}.$$

4. *La coünité de l'adjonction $\epsilon : \mathrm{Colim}_{\mathcal{C}} \circ I_{\mathcal{C}} \dot{\to} Id_{\mathcal{C}}$ est un isomorphisme naturel.*

Preuve. Ces résultats sont des conséquences immédiates de la définition de $\mathrm{Colim}_{\mathcal{C}}$.

1. Soit un diagramme $\overline{\alpha}$.

$$\begin{aligned} & I_{\mathcal{C}}(id_{\mathrm{Colim}_{\mathcal{C}}\,\overline{\alpha}}) \circ \overline{\eta}_{\overline{\alpha}} \\ &= \overline{id}_{I_{\mathcal{C}}(\mathrm{Colim}_{\mathcal{C}}\,\overline{\alpha})} \circ \overline{\eta}_{\overline{\alpha}} \quad (I_{\mathcal{C}} \text{ est un foncteur}) \\ &= \overline{\eta}_{\overline{\alpha}} \end{aligned}$$

Or, $\mathrm{Colim}_{\mathcal{C}}\,\overline{id}_{\overline{\alpha}}$ est l'unique flèche telle que $I_{\mathcal{C}}(\mathrm{Colim}_{\mathcal{C}}\,\overline{id}_{\overline{\alpha}}) \circ \overline{\eta}_{\overline{\alpha}} = \overline{\eta}_{\overline{\alpha}}$, donc

$$\mathrm{Colim}_{\mathcal{C}}\,\overline{id}_{\overline{\alpha}} = \overline{id}_{\mathrm{Colim}_{\mathcal{C}}\,\overline{\alpha}}.$$

Soit trois diagrammes $\overline{\alpha}$, $\overline{\beta}$ et $\overline{\gamma}$, et deux flèches $\overline{\sigma} : \overline{\alpha} \to \overline{\beta}$ et $\overline{\tau} : \overline{\beta} \to \overline{\gamma}$.

$$\begin{aligned} & I_{\mathcal{C}}(\mathrm{Colim}_{\mathcal{C}}\,\overline{\tau} \circ \mathrm{Colim}_{\mathcal{C}}\,\overline{\sigma}) \circ \overline{\eta}_{\overline{\alpha}} \\ &= I_{\mathcal{C}}(\mathrm{Colim}_{\mathcal{C}}\,\overline{\tau}) \circ I_{\mathcal{C}}(\mathrm{Colim}_{\mathcal{C}}\,\overline{\sigma}) \circ \overline{\eta}_{\overline{\alpha}} && (I_{\mathcal{C}} \text{ est un foncteur}) \\ &= I_{\mathcal{C}}(\mathrm{Colim}_{\mathcal{C}}\,\overline{\tau}) \circ \overline{\eta}_{\overline{\beta}} \circ \overline{\sigma} && (\text{définition de } \mathrm{Colim}_{\mathcal{C}}\,\overline{\sigma}) \\ &= \overline{\eta}_{\overline{\gamma}} \circ \overline{\tau} \circ \overline{\sigma} && (\text{définition de } \mathrm{Colim}_{\mathcal{C}}\,\overline{\tau}) \end{aligned}$$

Or, $\mathrm{Colim}_{\mathcal{C}}(\overline{\tau} \circ \overline{\sigma})$ est l'unique flèche telle que

$$\mathrm{Colim}_{\mathcal{C}}(\overline{\tau} \circ \overline{\sigma}) \circ \overline{\eta}_{\overline{\alpha}} = \overline{\eta}_{\overline{\gamma}} \circ \overline{\tau} \circ \overline{\sigma}$$

donc

$$\mathrm{Colim}_{\mathcal{C}}(\overline{\tau} \circ \overline{\sigma}) = \mathrm{Colim}_{\mathcal{C}}\,\overline{\tau} \circ \mathrm{Colim}_{\mathcal{C}}\,\overline{\sigma}.$$

Par conséquent, $\mathrm{Colim}_{\mathcal{C}} : \mathrm{DIAGR}(\mathcal{C}) \to \mathcal{C}$ est un foncteur.

2. Soit deux diagrammes $\overline{\alpha}$, $\overline{\beta}$ et une flèche $\overline{\sigma} : \overline{\alpha} \to \overline{\beta}$. Par définition de $\mathrm{Colim}_{\mathcal{C}}\,\overline{\sigma}$, on a bien

$$I_{\mathcal{C}}(\mathrm{Colim}_{\mathcal{C}}\,\overline{\sigma}) \circ \overline{\eta}_{\overline{\alpha}} = \overline{\eta}_{\overline{\beta}} \circ \overline{\sigma}.$$

3. Soit un diagramme $\overline{\alpha}$, un objet B de $\mathcal{C}$, et une flèche $\overline{\sigma} : \overline{\alpha} \to I_{\mathcal{C}}(B)$. Nous avons un cône $(B,\ \overline{\sigma} : \overline{\alpha} \to I_{\mathcal{C}}(B))$ sur le diagramme $\overline{\alpha}$, donc il existe une unique flèche $g : \mathrm{Colim}_{\mathcal{C}}\,\overline{\alpha} \to B$ de $\mathcal{C}$ telle que

$$I_{\mathcal{C}}(g) \circ \overline{\eta}_{\overline{\alpha}} = \overline{\sigma}.$$

4. D'après la propriété 2.2, il suffit de montrer que pour tout objet B de $\mathcal{C}$,

$$\epsilon_B : \mathrm{Colim}_{\mathcal{C}}\, I_{\mathcal{C}}(B) \to B$$

est un isomorphisme. Comme on a un adjonction $(\mathrm{Colim}_{\mathcal{C}} \dashv I_{\mathcal{C}})$, et d'après la propriété 2.4,

$$\begin{aligned} & I_{\mathcal{C}}\epsilon \cdot \eta I_{\mathcal{C}} = id_{I_{\mathcal{C}}} \\ & \Rightarrow \quad I_{\mathcal{C}}(\epsilon_B) \circ \overline{\eta}_{I_{\mathcal{C}}(B)} = \overline{id}_{I_{\mathcal{C}}(B)} \\ & \Rightarrow \quad \epsilon_B \circ (\eta_{I_{\mathcal{C}}(B)})_* = id_B. \end{aligned}$$

D'autre part,

$$\begin{aligned} & I_{\mathcal{C}}(\epsilon_B) \circ \overline{\eta}_{I_{\mathcal{C}}(B)} = \overline{id}_{I_{\mathcal{C}}(B)} \\ \Rightarrow \quad & \overline{\eta}_{I_{\mathcal{C}}(B)} \circ I_{\mathcal{C}}(\epsilon_B) \circ \overline{\eta}_{I_{\mathcal{C}}(B)} = \overline{\eta}_{I_{\mathcal{C}}(B)} \\ \Rightarrow \quad & I_{\mathcal{C}}((\eta_{I_{\mathcal{C}}(B)})_* \circ \epsilon_B) \circ \overline{\eta}_{I_{\mathcal{C}}(B)} = \overline{\eta}_{I_{\mathcal{C}}(B)} \quad (I_{\mathcal{C}} \text{ foncteur}) \end{aligned}$$

Comme $(\mathrm{Colim}_{\mathcal{C}}\, I_{\mathcal{C}}(B),\ \overline{\eta}_{I_{\mathcal{C}}(B)} : I_{\mathcal{C}}(B) \to I_{\mathcal{C}}(\mathrm{Colim}_{\mathcal{C}}\, I_{\mathcal{C}}(B)))$ est un cône colimite du diagramme $I_{\mathcal{C}}(B)$, il existe une unique flèche $\phi : \mathrm{Colim}_{\mathcal{C}}\, I_{\mathcal{C}}(B) \to \mathrm{Colim}_{\mathcal{C}}\, I_{\mathcal{C}}(B)$ de $\mathcal{C}$ telle que

$$I_{\mathcal{C}}(\phi) \circ \overline{\eta}_{I_{\mathcal{C}}(B)} = \overline{\eta}_{I_{\mathcal{C}}(B)}.$$

Or, les flèches $(\eta_{I_{\mathcal{C}}(B)})_* \circ \epsilon_B$ et $id_{\mathrm{Colim}_{\mathcal{C}}\, I_{\mathcal{C}}(B)}$ satisfont cette propriété, donc

$$(\eta_{I_{\mathcal{C}}(B)})_* \circ \epsilon_B = id_{\mathrm{Colim}_{\mathcal{C}}\, I_{\mathcal{C}}(B)}.$$

Donc $\epsilon_B : \mathrm{Colim}_{\mathcal{C}}\, I_{\mathcal{C}}(B) \to B$ est un isomorphisme, et $\epsilon_B^{-1} = (\eta_{I_{\mathcal{C}}(B)})_*$.

□

2.3.5 Conservation de colimites

Dans ce paragraphe, on considère deux catégories $\mathcal{C}$ et $\mathcal{D}$, et un foncteur $F : \mathcal{C} \to \mathcal{D}$. Nous définissons l'image d'un diagramme et d'un morphisme de diagrammes par ce foncteur: l'image d'un objet de DIAGR($\mathcal{C}$) est un objet de DIAGR($\mathcal{D}$), et l'image d'une flèche de DIAGR($\mathcal{C}$) est une flèche de DIAGR($\mathcal{D}$).

Définition 2.24 (Image d'un diagramme par un foncteur $F : \mathcal{C} \to \mathcal{D}$)
Soit un diagramme $\overline{\alpha} = (\alpha^\Phi,\ \alpha : \alpha^\Phi \to \mathcal{C})$ sur $\mathcal{C}$. L'image par F de $\overline{\alpha}$ est le diagramme sur $\mathcal{D}$

$$F \circ \overline{\alpha} = (\alpha^\Phi,\ F \circ \alpha : \alpha^\Phi \to \mathcal{D}).$$

Intuitivement, $F \circ \overline{\alpha}$ est un diagramme construit sur le graphe sous-jacent α^Φ du diagramme $\overline{\alpha}$. Les nœuds n de α^Φ sont étiquetés par les objets $F(\alpha(n))$ de $\mathcal{D}$, et les arcs $a : m \to n$ de α^Φ par les flèches $F(\alpha(a)) : F(\alpha(m)) \to F(\alpha(n))$ de $\mathcal{D}$.

Par exemple, étant donné un objet A de $\mathcal{C}$, le diagramme $F \circ I_{\mathcal{C}}(A)$ est le diagramme $I_{\mathcal{D}}(F(A))$.



diagramme $I_{\mathcal{C}}(A)$

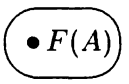


diagramme $F \circ I_{\mathcal{C}}(A) = I_{\mathcal{D}}(F(A))$

Lemme 2.3 *Soit un diagramme $\overline{\alpha}$ sur $\mathcal{C}$. Soit n_0 et n_k deux nœuds de α^Φ. Soit $u : A \to \alpha(n_0)$ et $v : A \to \alpha(n_k)$ deux flèches de $\mathcal{C}$ telles que*

$$u \sim_{\overline{\alpha}} v \ [Z : n_0 \rightsquigarrow n_k].$$

Alors,

$$F(u) \sim_{F\circ\overline{\alpha}} F(v) \ [Z : n_0 \rightsquigarrow n_k].$$

Remarquons que si $Z : n_0 \rightsquigarrow n_k$ est un zigzag sur le diagramme $\overline{\alpha}$, alors c'est également un zigzag sur le diagramme $F \circ \overline{\alpha}$, puisque ces deux diagrammes sont construits sur le même graphe sous-jacent α^Φ.

Preuve. Application immédiate des définitions de $F \circ \overline{\alpha}$, $\sim_{\overline{\alpha}}$ et $\sim_{F\circ\overline{\alpha}}$. □

Définition 2.25 (Image d'un morphisme de diagrammes par F)
Soit $\overline{\sigma} : \overline{\alpha} \to \overline{\beta} = (\sigma^\Phi : \alpha^\Phi \rightsquigarrow \beta^\Phi,\ \sigma : \alpha \rightsquigarrow\!\!\!\!\cdot\ \beta \circ \sigma^\Phi)$ une flèche de DIAGR($\mathcal{C}$).
L'image par F de $\overline{\sigma}$ est la flèche de DIAGR($\mathcal{D}$)

$$F\overline{\sigma} = (\sigma^\Phi : \alpha^\Phi \rightsquigarrow \beta^\Phi,\ F\sigma : F \circ \alpha \rightsquigarrow\!\!\!\!\cdot\ F \circ \beta \circ \sigma^\Phi)$$

où $F\sigma : F \circ \alpha \rightsquigarrow\!\!\!\!\cdot\ F \circ \beta \circ \sigma^\Phi$ est la transformation naturelle généralisée définie par

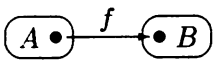
$$\forall n \in \text{Nœud}(\alpha^\Phi),\ (F\sigma)_n = F(\sigma_n).$$

On vérifie que cela définit bien une transformation naturelle généralisée. Considérons un arc $a : m \to n$ de α^Φ.

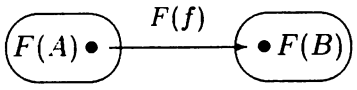
$\sigma_n \circ \alpha(a) \sim_{\overline{\beta}} \sigma_m \ [\sigma^\Phi(a)]$	(σ transformation nat. généralisée)
$\Rightarrow\ F(\sigma_n \circ \alpha(a)) \sim_{F\circ\overline{\beta}} F(\sigma_m) \ [\sigma^\Phi(a)]$	(lemme 2.3)
$\Rightarrow\ F(\sigma_n) \circ F(\alpha(a)) \sim_{F\circ\overline{\beta}} F(\sigma_m) \ [\sigma^\Phi(a)]$	(F foncteur)
$\Rightarrow\ (F\sigma)_n \circ F(\alpha(a)) \sim_{F\circ\overline{\beta}} (F\sigma)_n \ [\sigma^\Phi(a)]$	(définition de $F\sigma$)

Par conséquent, $F\sigma$ est bien une transformation naturelle généralisée.

Par exemple, soit une flèche $f : A \to B$ de $\mathcal{C}$. La flèche $FI_{\mathcal{C}}(f)$ est égale à $I_{\mathcal{D}}(F(f))$.



flèche $I_{\mathcal{C}}(f)$



morphisme de diagrammes $FI_{\mathcal{C}}(f) = I_{\mathcal{D}}(F(f))$

Pour tout diagramme $\overline{\alpha}$ de DIAGR($\mathcal{C}$), nous avons donc défini un diagramme $F \circ \overline{\alpha}$ de DIAGR($\mathcal{D}$) ; et pour toute flèche $\overline{\sigma} : \overline{\alpha} \to \overline{\beta}$ de DIAGR($\mathcal{C}$), nous avons défini une flèche $F\overline{\sigma} : F \circ \overline{\alpha} \to F \circ \overline{\beta}$ de DIAGR($\mathcal{D}$). Ces deux applications forment en fait un foncteur.

Lemme 2.4 *L'application*

$$\begin{array}{rccc} \text{DIAGR}(F) : & \text{DIAGR}(\mathcal{C}) & \to & \text{DIAGR}(\mathcal{D}) \\ & \overline{\alpha} & \mapsto & F \circ \overline{\alpha} \\ & \overline{\sigma} : \overline{\alpha} \to \overline{\beta} & \mapsto & F\overline{\sigma} : F \circ \overline{\alpha} \to F \circ \overline{\beta} \end{array}$$

est un foncteur DIAGR(F) : DIAGR($\mathcal{C}$) $\to$ DIAGR($\mathcal{D}$).

Finalement, nous avons donc défini DIAGR à la fois sur les catégories (objets de **CAT**) et sur les foncteurs (flèches de **CAT**). De nouveau, nous avons défini un foncteur.

Lemme 2.5 *L'application*

$$\begin{array}{rccc} \text{DIAGR} : & \mathbf{CAT} & \to & \mathbf{CAT} \\ & \mathcal{C} & \mapsto & \text{DIAGR}(\mathcal{C}) \\ & F : \mathcal{C} \to \mathcal{D} & \mapsto & \text{DIAGR}(F) : \text{DIAGR}(\mathcal{C}) \to \text{DIAGR}(\mathcal{D}) \end{array}$$

est un foncteur DIAGR : **CAT** $\to$ **CAT**.

Propriété 2.5 *L'application I qui à toute catégorie $\mathcal{C}$ associe le foncteur $I_{\mathcal{C}} : \mathcal{C} \to$ DIAGR($\mathcal{C}$) est une transformation naturelle*

$$I : Id_{\mathbf{CAT}} \dot{\to} \text{DIAGR}.$$

Preuve. Pour tout foncteur $F : \mathcal{C} \to \mathcal{D}$, on a

$$\text{DIAGR}(F) \circ I_{\mathcal{C}} = I_{\mathcal{D}} \circ F.$$

□

Lemme 2.6 *Soit deux foncteurs $F, F' : \mathcal{C} \to \mathcal{D}$. Alors,*

$$F \cong F' \Rightarrow \text{DIAGR}(F) \cong \text{DIAGR}(F').$$

Preuve. Soit $\omega : F \dot{\to} F'$ l'isomorphisme naturel entre ces deux foncteurs.
On définit une transformation naturelle

$$\overline{\xi} : \text{DIAGR}(F) \dot{\to} \text{DIAGR}(F').$$

À tout objet $\overline{\alpha}$ de DIAGR($\mathcal{C}$), on associe une flèche

$$\overline{\xi}_{\overline{\alpha}} : F \circ \overline{\alpha} \to F' \circ \overline{\alpha}$$

de DIAGR($\mathcal{D}$) définie la façon suivante.

- Les deux diagrammes $F \circ \overline{\alpha}$ et $F' \circ \overline{\alpha}$ sont construits sur le graphe α^{Φ} donc on peut définir $\xi^{\Phi}_{\overline{\alpha}} = id_{\alpha^{\Phi}}$.
- Pour tout nœud n de α^{Φ}, on pose

$$(\xi_{\overline{\alpha}})_n = \omega_{\alpha(n)} : F(\alpha(n)) \to F'(\alpha(n)).$$

Comme ω est une transformation naturelle, pour tout arc $a : m \to n$ de α^Φ, on a

$$\begin{aligned} & F'(\alpha(a)) \circ \omega_{\alpha(m)} = \omega_{\alpha(n)} \circ F(\alpha(a)) \\ \Leftrightarrow \quad & F'(\alpha(a)) \circ (\xi_{\overline{\alpha}})_m = (\xi_{\overline{\alpha}})_n \circ F(\alpha(a)) \end{aligned}$$

Par conséquent, $\xi_{\overline{\alpha}} : F \circ \overline{\alpha} \dot{\to} F' \circ \overline{\alpha}$ est une transformation naturelle, et donc $\overline{\xi}_{\overline{\alpha}} : F \circ \overline{\alpha} \to F' \circ \overline{\alpha}$ est bien une flèche de DIAGR($\mathcal{D}$).
De plus, $\overline{\xi}_{\overline{\alpha}}$ est un isomorphisme, et pour tout nœud n de α^Φ, $(\xi_{\overline{\alpha}}^{-1})_n = \omega_{\alpha(n)}^{-1}$.

On montre que

$$\overline{\xi} : \mathrm{DIAGR}(F) \dot{\to} \mathrm{DIAGR}(F')$$

est une transformation naturelle. Pour toute flèche $\overline{\sigma} : \overline{\alpha} \to \overline{\beta}$ de DIAGR($\mathcal{C}$), il faut vérifier que

$$\overline{\xi}_{\overline{\beta}} \circ \mathrm{DIAGR}(F)(\overline{\sigma}) = \mathrm{DIAGR}(F')(\overline{\sigma}) \circ \overline{\xi}_{\overline{\alpha}}.$$

Pour montrer cette égalité, on vérifie que les morphismes généralisés et les transformations naturelles généralisées qui correspondent aux deux membres coïncident. Notons

$$\begin{aligned} \overline{\tau} &= \overline{\xi}_{\overline{\beta}} \circ \mathrm{DIAGR}(F)(\overline{\sigma}) \\ \overline{\tau}' &= \mathrm{DIAGR}(F')(\overline{\sigma}) \circ \overline{\xi}_{\overline{\alpha}}. \end{aligned}$$

- $\tau^\Phi = \sigma^\Phi = \tau'^\Phi$.
- Pour tout nœud n de α^Φ,

$$\begin{aligned} \tau_n &= (\xi_{\overline{\beta}})_{\sigma^\Phi(n)} \circ F(\sigma_n) && \text{(définition de } \overline{\tau}) \\ &= \omega_{\beta(\sigma^\Phi(n))} \circ F(\sigma_n) && \text{(définition de } \overline{\xi}) \\ &= F'(\sigma_n) \circ \omega_{\alpha(n)} && (\omega \text{ transformation naturelle}) \\ &= F'(\sigma_n) \circ (\xi_{\overline{\alpha}})_n && \text{(définition de } \overline{\xi}) \\ &= \tau'_n && \text{(définition de } \overline{\tau}') \end{aligned}$$

Par conséquent, on a bien un isomorphisme naturel

$$\overline{\xi} : \mathrm{DIAGR}(F) \dot{\to} \mathrm{DIAGR}(F').$$

□

Définition 2.26 (Foncteur conservant une colimite)
Soit un diagramme $\overline{\alpha} = (\alpha^\Phi,\ \alpha : \alpha^\Phi \to \mathcal{C})$, ayant pour cône colimite

$$(A,\ \overline{\eta}_{\overline{\alpha}} : \overline{\alpha} \to I_{\mathcal{C}}(A)).$$

On dit que le foncteur $F : \mathcal{C} \to \mathcal{D}$ *conserve la colimite de* $\overline{\alpha}$ si et seulement si

$$(F(A),\ \mathrm{DIAGR}(F)(\overline{\eta}_{\overline{\alpha}}) : F \circ \overline{\alpha} \to I_{\mathcal{D}}(F(A)))$$

est un cône colimite du diagramme $F \circ \overline{\alpha}$.

Définition 2.27 (Foncteur conservant les colimites finies)
Soit deux catégories finiment cocomplètes $\mathcal{C}$ et $\mathcal{D}$. On dit qu'un foncteur $F : \mathcal{C} \to \mathcal{D}$ *conserve les colimites finies* si et seulement si pour tout diagramme fini $\overline{\alpha}$ sur $\mathcal{C}$, F conserve la colimite de $\overline{\alpha}$.

Propriété 2.6 *Soit deux catégories finiment cocomplètes $\mathcal{C}$ et $\mathcal{D}$, et un foncteur $F : \mathcal{C} \to \mathcal{D}$ qui conserve les colimites finies. Alors, il existe un isomorphisme naturel*

$$\mathrm{Colim}_{\mathcal{D}} \circ \mathrm{DIAGR}(F) \cong F \circ \mathrm{Colim}_{\mathcal{C}}.$$

Preuve. Soit un diagramme $\overline{\alpha}$ sur $\mathcal{C}$, ayant pour cône colimite

$$(\mathrm{Colim}_{\mathcal{C}}\, \overline{\alpha},\ \overline{\eta_{\overline{\alpha}}} : \overline{\alpha} \to I_{\mathcal{C}}(\mathrm{Colim}_{\mathcal{C}}\, \overline{\alpha})).$$

On considère le diagramme $F \circ \overline{\alpha}$ sur $\mathcal{D}$, qui a pour cône colimite

$$(\mathrm{Colim}_{\mathcal{D}}(F \circ \overline{\alpha}),\ \overline{\eta}_{F \circ \overline{\alpha}} : F \circ \overline{\alpha} \to I_{\mathcal{D}}(\mathrm{Colim}_{\mathcal{D}}(F \circ \overline{\alpha}))).$$

Comme F conserve la colimite de $\overline{\alpha}$, et d'après le lemme 2.1,

$$\mathrm{Colim}_{\mathcal{D}}(F \circ \overline{\alpha}) \cong F(\mathrm{Colim}_{\mathcal{C}}\, \overline{\alpha})$$

et l'isomorphisme

$$\phi_{\overline{\alpha}} : \mathrm{Colim}_{\mathcal{D}}(F \circ \overline{\alpha}) \to F(\mathrm{Colim}_{\mathcal{C}}\, \overline{\alpha})$$

est l'unique flèche telle que $I_{\mathcal{D}}(\phi_{\overline{\alpha}}) \circ \overline{\eta}_{F \circ \overline{\alpha}} = \mathrm{DIAGR}(F)(\overline{\eta_{\overline{\alpha}}})$.
Il suffit de montrer qu'on a une transformation naturelle

$$\phi : \mathrm{Colim}_{\mathcal{D}} \circ \mathrm{DIAGR}(F) \dot{\to} F \circ \mathrm{Colim}_{\mathcal{C}}.$$

Soit un diagramme $\overline{\beta}$, ayant pour cône colimite

$$(\mathrm{Colim}_{\mathcal{C}}\, \overline{\beta},\ \overline{\eta_{\overline{\beta}}} : \overline{\beta} \to I_{\mathcal{C}}(\mathrm{Colim}_{\mathcal{C}}\, \overline{\beta})).$$

Soit

$$\phi_{\overline{\beta}} : \mathrm{Colim}_{\mathcal{D}}(F \circ \overline{\beta}) \to F(\mathrm{Colim}_{\mathcal{C}}\, \overline{\beta})$$

l'unique flèche telle que

$$I_{\mathcal{D}}(\phi_{\overline{\beta}}) \circ \overline{\eta}_{F \circ \overline{\beta}} = \mathrm{DIAGR}(F)(\overline{\eta_{\overline{\beta}}}).$$

Soit $\overline{\sigma} : \overline{\alpha} \to \overline{\beta}$ une flèche de $\mathrm{DIAGR}(\mathcal{C})$. Pour montrer que ϕ est une transformation naturelle, il faut montrer que

$$F(\mathrm{Colim}_{\mathcal{C}}\, \overline{\sigma}) \circ \phi_{\overline{\alpha}} = \phi_{\overline{\beta}} \circ \mathrm{Colim}_{\mathcal{D}}(\mathrm{DIAGR}(F)(\overline{\sigma})).$$

Pour cela, il suffit de montrer que

$$I_{\mathcal{D}}(F(\mathrm{Colim}_{\mathcal{C}}\,\overline{\sigma})\circ\phi_{\overline{\alpha}})\circ\overline{\eta}_{F\circ\overline{\alpha}} = I_{\mathcal{D}}(\phi_{\overline{\beta}}\circ\mathrm{Colim}_{\mathcal{D}}(\mathrm{DIAGR}(F)(\overline{\sigma})))\circ\overline{\eta}_{F\circ\overline{\alpha}}.$$

$$\begin{array}{lll}
I_{\mathcal{D}}(F(\mathrm{Colim}_{\mathcal{C}}\,\overline{\sigma})\circ\phi_{\overline{\alpha}})\circ\overline{\eta}_{F\circ\overline{\alpha}} & & \\
= & I_{\mathcal{D}}(F(\mathrm{Colim}_{\mathcal{C}}\,\overline{\sigma}))\circ I_{\mathcal{D}}(\phi_{\overline{\alpha}})\circ\overline{\eta}_{F\circ\overline{\alpha}} & (I_{\mathcal{D}}\text{ est un foncteur}) \\
= & I_{\mathcal{D}}(F(\mathrm{Colim}_{\mathcal{C}}\,\overline{\sigma}))\circ\mathrm{DIAGR}(F)(\overline{\eta}_{\overline{\alpha}}) & (\text{définition de } \phi_{\overline{\alpha}}) \\
= & \mathrm{DIAGR}(F)(I_{\mathcal{C}}(\mathrm{Colim}_{\mathcal{C}}\,\overline{\sigma}))\circ\mathrm{DIAGR}(F)(\overline{\eta}_{\overline{\alpha}}) & (I : Id_{\mathbf{CAT}} \dot{\to} \mathrm{DIAGR}) \\
= & \mathrm{DIAGR}(F)(I_{\mathcal{C}}(\mathrm{Colim}_{\mathcal{C}}\,\overline{\sigma})\circ\overline{\eta}_{\overline{\alpha}}) & (\mathrm{DIAGR}(F)\text{ est un foncteur}) \\
= & \mathrm{DIAGR}(F)(\overline{\eta}_{\overline{\beta}}\circ\overline{\sigma}) & (\text{définition de } \mathrm{Colim}_{\mathcal{C}}\,\overline{\sigma}) \\
= & \mathrm{DIAGR}(F)(\overline{\eta}_{\overline{\beta}})\circ\mathrm{DIAGR}(F)(\overline{\sigma}) & (\mathrm{DIAGR}(F)\text{ est un foncteur}) \\
= & I_{\mathcal{D}}(\phi_{\overline{\beta}})\circ\overline{\eta}_{F\circ\overline{\beta}}\circ\mathrm{DIAGR}(F)(\overline{\sigma}) & (\text{définition de } \phi_{\overline{\beta}}) \\
= & I_{\mathcal{D}}(\phi_{\overline{\beta}})\circ I_{\mathcal{D}}(\mathrm{Colim}_{\mathcal{D}}(\mathrm{DIAGR}(F)(\overline{\sigma})))\circ\overline{\eta}_{F\circ\overline{\alpha}} & (\text{déf. } \mathrm{Colim}_{\mathcal{D}}(\mathrm{DIAGR}(F)(\overline{\sigma}))) \\
= & I_{\mathcal{D}}(\varphi_{\overline{\beta}}\circ\mathrm{Colim}_{\mathcal{D}}(\mathrm{DIAGR}(F)(\overline{\sigma})))\circ\overline{\eta}_{F\circ\overline{\alpha}} & (I_{\mathcal{D}}\text{ est un foncteur})
\end{array}$$

Par conséquent, on a bien une transformation naturelle

$$\varphi : \mathrm{Colim}_{\mathcal{D}}\circ\mathrm{DIAGR}(F) \dot{\to} F\circ\mathrm{Colim}_{\mathcal{C}}.$$

Comme pour tout diagramme $\overline{\alpha}$, $\phi_{\overline{\alpha}}$ est un isomorphisme, on a donc un isomorphisme naturel

$$\mathrm{Colim}_{\mathcal{D}}\circ\mathrm{DIAGR}(F) \cong F\circ\mathrm{Colim}_{\mathcal{C}}.$$

□

Si on fait certains choix de colimites dans les catégories $\mathcal{C}$ et $\mathcal{D}$, on peut renforcer la définition 2.26 en imposant que *la colimite choisie* du diagramme $\overline{\alpha}$ sur $\mathcal{C}$ soit envoyée sur *la colimite choisie* du diagramme $F\circ\overline{\alpha}$ sur $\mathcal{D}$.

Définition 2.28 (Foncteur conservant fortement une colimite)
Soit un diagramme $\overline{\alpha} = (\alpha^{\Phi},\ \alpha : \alpha^{\Phi} \to \mathcal{C})$, ayant pour cône colimite choisi

$$(A,\ \overline{\eta}_{\overline{\alpha}} : \overline{\alpha} \to I_{\mathcal{C}}(A)).$$

On dit que le foncteur $F : \mathcal{C} \to \mathcal{D}$ *conserve fortement la colimite choisie de* $\overline{\alpha}$ si et seulement si

$$(F(A),\ \mathrm{DIAGR}(F)(\overline{\eta}_{\overline{\alpha}}) : F\circ\overline{\alpha} \to I_{\mathcal{D}}(F(A)))$$

est le cône colimite choisi du diagramme $F\circ\overline{\alpha}$ sur $\mathcal{D}$.

Si F conserve fortement la colimite choisie de $\overline{\alpha}$, on a donc

$$F(\mathrm{Colim}_{\mathcal{C}}\,\overline{\alpha}) = \mathrm{Colim}_{\mathcal{D}}(F\circ\overline{\alpha}).$$

2.3.6 Aplatissement

Diagrammes de diagrammes : catégorie $\mathrm{DIAGR}^2(\mathcal{C})$

Étant donné une catégorie $\mathcal{C}$, on peut considérer la catégorie

$$\mathrm{DIAGR}^2(\mathcal{C}) = \mathrm{DIAGR}(\mathrm{DIAGR}(\mathcal{C}))$$

qui a pour objets des "diagrammes de diagrammes", c'est-à-dire des diagrammes sur $\mathrm{DIAGR}(\mathcal{C})$. Un objet $\overline{\overline{\Delta}}$ de $\mathrm{DIAGR}^2(\mathcal{C})$ est donc un couple

$$\overline{\overline{\Delta}} = (\Delta^\Phi,\ \Delta : \Delta^\Phi \to \mathrm{DIAGR}(\mathcal{C}))$$

où Δ^Φ est un graphe, et $\Delta : \Delta^\Phi \to \mathrm{DIAGR}(\mathcal{C})$ est un foncteur. Détaillons l'action de ce foncteur sur les nœuds et les arcs de Δ^Φ.

- Pour tout nœud N du graphe Δ^Φ, on a un diagramme sur $\mathcal{C}$

$$\overline{\Delta(N)} = (\Delta(N)^\Phi,\ \Delta(N) : \Delta(N)^\Phi \to \mathcal{C})$$

où

  - $\Delta(N)^\Phi$ est un graphe ;
  - $\Delta(N) : \Delta(N)^\Phi \to \mathcal{C}$ est un foncteur, qui associe à tout nœud n du graphe $\Delta(N)^\Phi$ un objet $\Delta(N)(n)$ de $\mathcal{C}$, et à tout arc $a : n \to n'$ du graphe $\Delta(N)^\Phi$ une flèche $\Delta(N)(a)$ de $\mathcal{C}$.

- Pour tout arc $A : N \to N'$ du graphe Δ^Φ, on a un morphisme de diagrammes $\overline{\Delta(A)} : \overline{\Delta(N)} \to \overline{\Delta(N')}$ sur $\mathcal{C}$

$$\overline{\Delta(A)} = (\Delta(A)^\Phi : \Delta(N)^\Phi \rightsquigarrow \Delta(N')^\Phi,\ \Delta(A) : \Delta(N) \dashrightarrow \Delta(N') \circ \Delta(A)^\Phi),$$

où

  - $\Delta(A)^\Phi : \Delta(N)^\Phi \rightsquigarrow \Delta(N')^\Phi$ est un morphisme généralisé de graphes, qui associe à tout arc $a : n \to n'$ de $\Delta(N)^\Phi$ un zigzag $\Delta(A)^\Phi(a) : \Delta(A)^\Phi(n) \rightsquigarrow \Delta(A)^\Phi(n')$ du graphe $\Delta(N')^\Phi$;
  - $\Delta(A) : \Delta(N) \dashrightarrow \Delta(N') \circ \Delta(A)^\Phi$ est une transformation naturelle généralisée, qui associe à tout nœud n de $\Delta(N)^\Phi$ une flèche $\Delta(A)_n : \Delta(N)(n) \to \Delta(N')(\Delta(A)^\Phi(n))$ de $\mathcal{C}$.

Aplatissement $\mathrm{Apl}_\mathcal{C} : \mathrm{DIAGR}^2(\mathcal{C}) \to \mathrm{DIAGR}(\mathcal{C})$

Étant donné une catégorie $\mathcal{C}$, le foncteur $I_\mathcal{C} : \mathcal{C} \to \mathrm{DIAGR}(\mathcal{C})$ permet de créer un objet de $\mathrm{DIAGR}(\mathcal{C})$ à partir d'un objet de $\mathcal{C}$, et une flèche de $\mathrm{DIAGR}(\mathcal{C})$ à partir d'une flèche de $\mathcal{C}$. La fonction d'aplatissement

$$\mathrm{Apl}_\mathcal{C} : \mathrm{DIAGR}^2(\mathcal{C}) \to \mathrm{DIAGR}(\mathcal{C})$$

permet au contraire de "détruire de la structure", en transformant tout diagramme de diagrammes (c'est-à-dire tout objet de $\mathrm{DIAGR}^2(\mathcal{C})$) en un simple diagramme (c'est-à-dire en un objet de $\mathrm{DIAGR}(\mathcal{C})$).

Considérons par exemple le diagramme $\overline{\overline{\Delta}}$ sur $\mathrm{DIAGR}(\mathcal{C})$, figure 2.2.

$$\overline{\overline{\Delta}} = (\Delta^{\Phi},\ \Delta : \Delta^{\Phi} \to \mathrm{DIAGR}(\mathcal{C}))$$

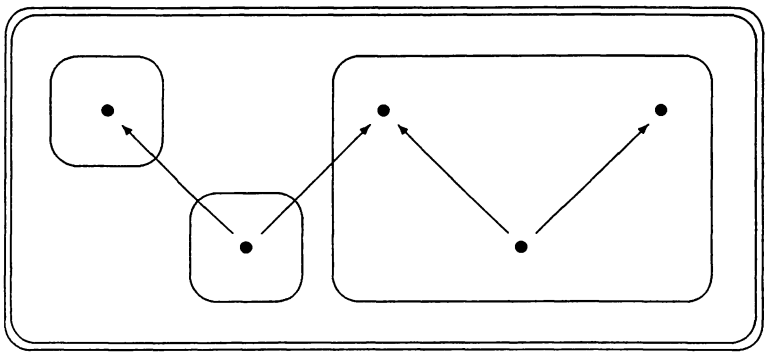


FIG. 2.2 – : *diagramme* $\overline{\overline{\Delta}}$, *objet de* $\mathrm{DIAGR}^2(\mathcal{C})$

Intuitivement, aplatir le diagramme de diagrammes $\overline{\overline{\Delta}}$ consiste à faire la réunion de tous les sous diagrammes de $\overline{\overline{\Delta}}$, et à transformer toutes les flèches de $\mathrm{DIAGR}(\mathcal{C})$ en (ensembles de) flèches de $\mathcal{C}$ (cf. figure 2.3).

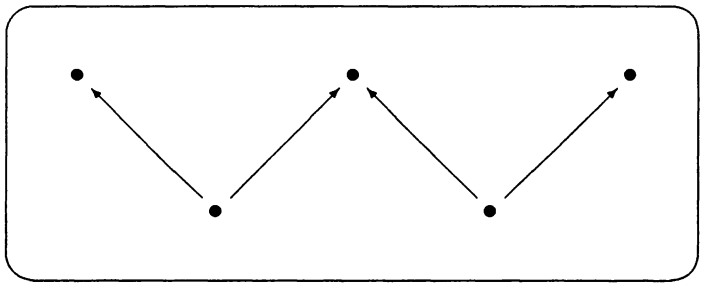


FIG. 2.3 – : *Aplatissement du diagramme* $\overline{\overline{\Delta}}$
$\overline{\delta} = \mathrm{Apl}_{\mathcal{C}}(\overline{\overline{\Delta}})$, *objet de* $\mathrm{DIAGR}(\mathcal{C})$

Définition 2.29 (Aplatissement $\mathrm{Apl}_{\mathcal{C}} : \mathrm{DIAGR}^2(\mathcal{C}) \to \mathrm{DIAGR}(\mathcal{C})$)
Soit

$$\overline{\overline{\Delta}} = (\Delta^{\Phi},\ \Delta : \Delta^{\Phi} \to \mathrm{DIAGR}(\mathcal{C}))$$

un objet de $\mathrm{DIAGR}^2(\mathcal{C})$. On définit le diagramme $\overline{\delta} = \mathrm{Apl}_{\mathcal{C}}(\overline{\overline{\Delta}})$ de $\mathrm{DIAGR}(\mathcal{C})$

$$\mathrm{Apl}_{\mathcal{C}}(\overline{\overline{\Delta}}) = \overline{\delta} = (\delta^{\Phi},\ \delta : \delta^{\Phi} \to \mathcal{C})$$

de la façon suivante.

- δ^Φ est un graphe, déterminé par l'ensemble de ses nœuds et l'ensemble de ses arcs.

$$\text{Nœud}(\delta^\Phi) = \{\ (N,n)\ ;\ N \in \text{Nœud}(\Delta^\Phi)\ ;\ n \in \text{Nœud}(\Delta(N)^\Phi)\ \}$$

$$\begin{aligned}\text{Arc}(\delta^\Phi) \quad &= \{\ (N,a) : (N,n) \to (N,n')\ ; \\ &\qquad N \in \text{Nœud}(\Delta^\Phi)\ ; \\ &\qquad n, n' \in \text{Nœud}(\Delta(N)^\Phi)\ ; \\ &\qquad a : n \to n' \in \text{Arc}(\Delta(N)^\Phi)\ \} \\ &\cup \{\ (A,n) : (N,n) \to (N', \Delta(A)^\Phi(n))\ ; \\ &\qquad N, N' \in \text{Nœud}(\Delta^\Phi)\ ; \\ &\qquad A : N \to N' \in \text{Arc}(\Delta^\Phi)\ ; \\ &\qquad n \in \text{Nœud}(\Delta(N)^\Phi)\ \}\end{aligned}$$

- $\delta : \delta^\Phi \to \mathcal{C}$ est un foncteur, défini sur les nœuds et arcs de δ^Φ de la façon suivante.

  - Action sur les nœuds :

$$\delta(N,n) = \Delta(N)(n)$$

  - Action sur les arcs :

$$\begin{aligned}\delta(N,a) &= \Delta(N)(a) \\ \delta(A,n) &= \Delta(A)_n\end{aligned}$$

Définition 2.30 (Ensemble de flèches $\{\overline{J_N} : \overline{\Delta(N)} \to \overline{\delta},\ \forall N \in \text{Nœud}(\Delta^\Phi)\}$)
Pour tout nœud N de Δ^Φ, on définit une flèche $\overline{J_N} : \overline{\Delta(N)} \to \overline{\delta}$ de DIAGR($\mathcal{C}$)

$$\overline{J_N} : \overline{\Delta(N)} \to \overline{\delta} = (J_N^\Phi : \Delta(N)^\Phi \rightsquigarrow \delta^\Phi,\ J_N : \Delta(N) \rightsquigarrow\!\!\!\!\cdot\ \ \delta \circ J_N^\Phi)$$

Le foncteur généralisé $J_N^\Phi : \Delta(N)^\Phi \rightsquigarrow \delta^\Phi$ est défini par

$$\begin{array}{rccc} J_N^\Phi\ : & \Delta(N)^\Phi & \rightsquigarrow & \delta^\Phi \\ & n & \mapsto & (N,n) \\ & a : n \to n' & \mapsto & (N,a) : (N,n) \to (N,n') \end{array}$$

Ce foncteur généralisé est en réalité un foncteur, car à tout arc de $\Delta(N)^\Phi$ est associé un *arc* de δ^Φ (et non un zigzag quelconque).
La transformation naturelle généralisée $J_N : \Delta(N) \rightsquigarrow\!\!\!\!\cdot\ \ \delta \circ J_N^\Phi$ associe à tout nœud n de $\Delta(N)^\Phi$ la flèche de $\mathcal{C}$

$$(J_N)_n = id_{\Delta(N)(n)} = id_{\delta(N,n)}.$$

Pour montrer que $\overline{J_N} : \overline{\Delta(N)} \to \overline{\delta}$ est bien une flèche de DIAGR($\mathcal{C}$), et comme J_N^Φ est un foncteur, il suffit de vérifier que J_N est une transformation naturelle

$$J_N : \Delta(N) \dot{\to} \delta \circ J_N^\Phi.$$

C'est le cas, car pour tout arc $a : n \to n'$ de $\Delta(N)^\Phi$, par définition de $\delta(N, a)$, on a

$$(J_N)_n \circ \Delta(N)(a) = \delta(N, a) \circ (J_N)_{n'}.$$

Remarque 2.6 Nous avons donc une famille de flèches

$$\overline{J_N} : \overline{\Delta(N)} \to \overline{\delta}, \ \forall N \in \text{Nœud}(\Delta^\Phi).$$

Pour tout nœud n de $\Delta(N)^\Phi$ on a donc une flèche

$$(J_N)_n : \Delta(N)(n) \to \delta(N, n).$$

Comme on a défini $\delta(N, n)$ égal à $\Delta(N)(n)$, cela est cohérent avec la définition de départ

$$(J_N)_n = id_{\Delta(N)(n)} = id_{\delta(N,n)}.$$

En fait, il n'est pas toujours commode de confondre les objets $\Delta(N)(n)$ et $\delta(N, n)$, parce que lors des raisonnements, le diagramme $\overline{\Delta(N)}$ ou $\overline{\delta}$ est en fait aussi important que l'objet $\delta(N, n) = \Delta(N)(n)$ de $\mathcal{C}$.
Pour cette raison, pour tout arc $a : n \to n'$ de $\Delta(N)^\Phi$, nous distinguerons les deux flèches

$$\Delta(N)(a) : \Delta(N)(n) \to \Delta(N)(n'),$$
$$\delta(N, a) : \delta(N, n) \to \delta(N, n').$$

De même, pour tout arc $A : N \to N'$ de Δ^Φ, et pour tout nœud n de $\Delta(N)^\Phi$, nous distinguerons les deux flèches

$$\Delta(A)_n : \Delta(N)(n) \to \Delta(N')(\Delta(A)^\Phi(n)),$$
$$\delta(A, n) : \delta(N, n) \to \delta(N', \Delta(A)^\Phi(n)).$$

Enfin, pour tout nœud N de Δ^Φ, et tout nœud n de $\Delta(N)^\Phi$, nous distinguerons également les trois flèches

$$(J_N)_n : \Delta(N)(n) \to \delta(N, n),$$
$$id_{\Delta(N)(n)} : \Delta(N)(n) \to \Delta(N)(n),$$
$$id_{\delta(N,n)} : \delta(N, n) \to \delta(N, n).$$

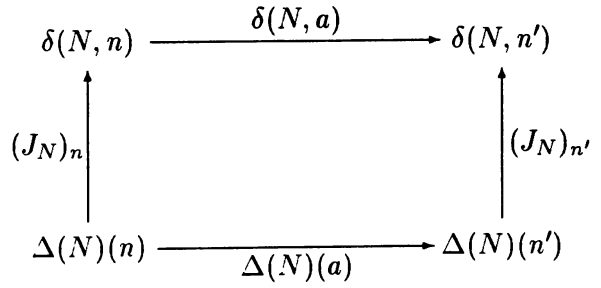
Pour respecter ces contraintes, nous n'utiliserons donc pas les deux égalités suivantes

$$\Delta(N)(a) = \delta(N, a),$$
$$\Delta(A)_n = \delta(A, n).$$

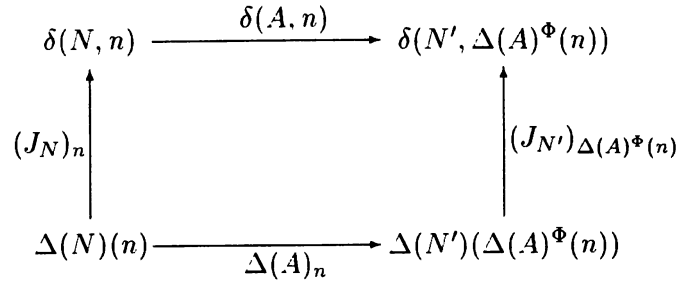
À la place, nous utiliserons les deux propriétés équivalentes suivantes.

Propriété 2.7

1. $\forall N \in \text{Nœud}(\Delta^\Phi),\ \forall a : n \to n' \in \text{Arc}(\Delta(N)^\Phi),$
$$(J_N)_{n'} \circ \Delta(N)(a) = \delta(N,a) \circ (J_N)_n.$$



2. $\forall A : N \to N' \in \text{Arc}(\Delta^\Phi),\ \forall n \in \text{Nœud}(\Delta(N)^\Phi),$
$$(J_{N'})_{\Delta(A)^\Phi(n)} \circ \Delta(A)_n = \delta(A,n) \circ (J_N)_n.$$



Remarque 2.7 La famille de flèches de $\text{DIAGR}(\mathcal{C})$
$$\overline{J_N} : \overline{\Delta(N)} \to \overline{\delta},\ \forall N \in \text{Nœud}(\Delta^\Phi)$$
ne permet pas de construire une flèche
$$\overline{\overline{J}} : \overline{\overline{\Delta}} \to I_{\text{DIAGR}(\mathcal{C})}(\overline{\delta})$$
dans la catégorie $\text{DIAGR}^2(\mathcal{C})$. En effet, il faudrait pour cela que pour tout arc $A : N \to N'$ de Δ^Φ, on ait
$$\overline{J_{N'}} \circ \overline{\Delta(A)} = \overline{J_N}$$
$$\Leftrightarrow\ \forall n \in \text{Nœud}(\Delta(N)^\Phi),\ (J_{N'})_{\Delta(A)^\Phi(n)} \circ \Delta(A)_n = (J_N)_n,$$
ce qui n'est pas vérifié.

Lemme 2.7 *Pour tout diagramme $\overline{\alpha}$ de* $\text{DIAGR}(\mathcal{C})$, *on a*
$$\text{Apl}_{\mathcal{C}}(I_{\text{DIAGR}(\mathcal{C})}(\overline{\alpha})) = \overline{\alpha}.$$

Preuve. Posons $\overline{\overline{\Delta}} = I_{\mathrm{DIAGR}(\mathcal{C})}(\overline{\alpha})$ et $\overline{\delta} = \mathrm{Apl}_{\mathcal{C}}(\overline{\overline{\Delta}})$. Le graphe δ^{Φ} est défini par

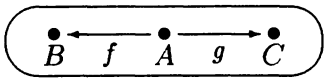
$$\mathrm{Nœud}(\delta^{\Phi}) = \{(*, n),\ \forall n \in \mathrm{Nœud}(\alpha^{\Phi})\}$$
$$\mathrm{Arc}(\delta^{\Phi}) = \{(*, a) : (*, n) \to (*, n'),\ \forall a : n \to n' \in \mathrm{Arc}(\alpha^{\Phi})\}$$

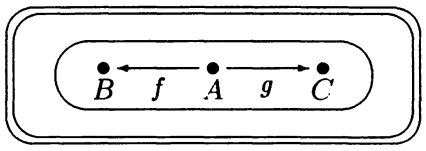
donc les graphes α^{Φ} et δ^{Φ} sont isomorphes. D'autre part,

$$\forall n \in \mathrm{Nœud}(\alpha^{\Phi}),\ \delta(*, n) = \Delta(*)(n) = \alpha(n)$$
$$\forall a : n \to n' \in \mathrm{Arc}(\alpha^{\Phi}),\ \delta(*, a) = \Delta(*)(a) = \alpha(a).$$

Par conséquent, $\overline{\alpha} = \overline{\delta}$. □

Donnons un exemple. On considère le diagramme $\overline{\alpha}$ suivant.

$\overline{\alpha} =$ 

$I_{\mathrm{DIAGR}(\mathcal{C})}(\overline{\alpha}) =$ 

Il est clair que l'aplatissement de $I_{\mathrm{DIAGR}(\mathcal{C})}(\overline{\alpha})$ donne le diagramme $\overline{\alpha}$.

Lemme 2.8 *Pour tout diagramme* $\overline{\alpha}$ *de* $\mathrm{DIAGR}(\mathcal{C})$, *on a*

$$\mathrm{Apl}_{\mathcal{C}}(\mathrm{DIAGR}(I_{\mathcal{C}})(\overline{\alpha})) = \overline{\alpha}.$$

Preuve. Posons $\overline{\overline{\Delta}} = \mathrm{DIAGR}(I_{\mathcal{C}})(\overline{\alpha}) = I_{\mathcal{C}} \circ \overline{\alpha}$, et $\overline{\delta} = \mathrm{Apl}_{\mathcal{C}}(\overline{\overline{\Delta}})$. Le graphe δ^{Φ} est défini par

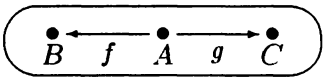
$$\mathrm{Nœud}(\delta^{\Phi}) = \{(n, *),\ \forall n \in \mathrm{Nœud}(\alpha^{\Phi})\}$$
$$\mathrm{Arc}(\delta^{\Phi}) = \{(a, *) : (n, *) \to (n', *),\ \forall a : n \to n' \in \mathrm{Arc}(\alpha^{\Phi})\}$$

donc les graphes α^{Φ} et δ^{Φ} sont isomorphes. D'autre part,

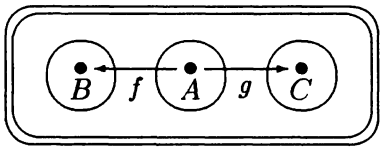
$$\delta(n, *) = \Delta(n)(*) = I_{\mathcal{C}}(\alpha(n))(*) = \alpha(n)$$
$$\delta(a, *) = \Delta(a)_{*} = I_{\mathcal{C}}(\alpha(a))_{*} = \alpha(a).$$

Par conséquent, $\overline{\alpha} = \overline{\delta}$. □

Illustrons ce résultat par un exemple. On considère le diagramme

$\overline{\alpha} =$ 

On a donc

$\mathrm{DIAGR}(I_{\mathcal{C}})(\overline{\alpha}) = I_{\mathcal{C}} \circ \overline{\alpha} =$ 

L'aplatissement de $I_{\mathcal{C}} \circ \overline{\alpha}$ est bien égal au diagramme $\overline{\alpha}$.

2.3.7 Catégorie diagr($\mathcal{C}$)

Dans la catégorie DIAGR($\mathcal{C}$), des morphismes de diagrammes différents peuvent avoir des colimites égales. Dans l'exemple 2.6, nous avons défini un morphisme de diagrammes $\overline{\sigma} : \overline{\alpha} \to \overline{\delta_2}$ tel que

$$\text{Colim}\,\overline{\sigma} = \mathsf{up}\,(S, M, G, A_2, b \circ s, m \circ b \circ s, \&_1(A_2) \circ \&_1(MD), \&_2(A_2)).$$

La définition de $\overline{\sigma}$ que nous avons donnée n'était pas la seule possible. Considérons par exemple le morphisme de diagrammes $\overline{\tau} : \overline{\alpha} \to \overline{\delta_2}$ suivant.

- Morphisme généralisé de graphes τ^Φ :

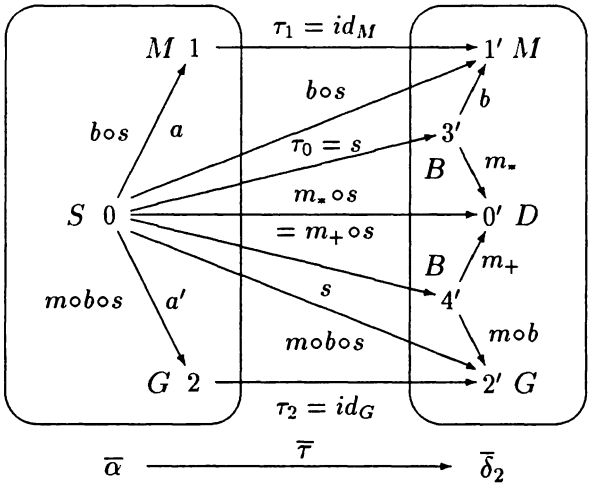
$$\begin{array}{rcl} \tau^\Phi : \alpha^\Phi & \rightsquigarrow & \delta_2^\Phi \\ 0 & \mapsto & 3' \\ 1 & \mapsto & 1' \\ 2 & \mapsto & 2' \\ a & \mapsto & b \\ a' & \mapsto & 3' \xrightarrow{m_*} 0' \xleftarrow{m_+} 4' \xrightarrow{mob} 2' \end{array}$$

- Transformation naturelle $\tau : \alpha \dot{\rightsquigarrow} \delta_2 \circ \tau^\Phi$:

$$\begin{aligned} \tau_0 &= s \\ \tau_1 &= id_M \\ \tau_2 &= id_G. \end{aligned}$$

Les colimites des morphismes de diagrammes $\overline{\sigma}$ et $\overline{\tau}$ sont égales.

$$\text{Colim}\,\overline{\tau} = \text{Colim}\,\overline{\sigma} = \mathsf{up}\,(S, M, G, A_2, b \circ s, m \circ b \circ s, \&_1(A_2) \circ \&_1(MD), \&_2(A_2)).$$

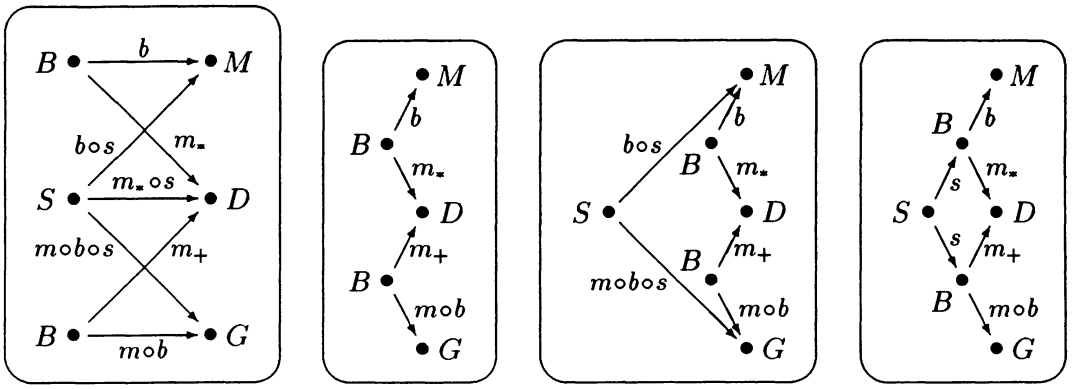


En fait, on peut choisir pour la flèche τ_0 une flèche quelconque du "faisceau" de flèches

$$\begin{array}{lcl} b \circ s & : & \alpha^\Phi(0) \to \delta_2^\Phi(1') \\ s & : & \alpha^\Phi(0) \to \delta_2^\Phi(3') \\ m_* \circ s = m_+ \circ s & : & \alpha^\Phi(0) \to \delta_2^\Phi(0') \\ s & : & \alpha^\Phi(0) \to \delta_2^\Phi(4') \\ m \circ b \circ s & : & \alpha^\Phi(0) \to \delta_2^\Phi(2'). \end{array}$$

Il faut évidemment définir τ^{Φ} en conséquence.

De même, deux diagrammes non isomorphes dans la catégorie DIAGR($\mathcal{C}$) peuvent avoir des colimites isomorphes. Par exemple, on peut vérifier que les diagrammes $\overline{\delta_1}$, $\overline{\delta_2}$, $\overline{\delta_3}$ et $\overline{\delta_4}$ définis dans le chapitre 1 (section 1.5) ont des colimites isomorphes.



Pour prouver que les colimites de ces quatre diagrammes sont isomorphes, il faut bien sûr utiliser les propriétés des colimites, ainsi que l'égalité entre flèches de $\mathcal{C}_0$:

$$m_+ \circ s = m_* \circ s.$$

Le but de ce paragraphe est de définir une catégorie dans laquelle les égalités entre flèches colimites et les isomorphismes entre objets colimites seront reflétées au niveau des diagrammes. Pour cela, nous définissons une relation de congruence $\approx$ sur les flèches de DIAGR($\mathcal{C}$), compatible avec les colimites, et diagr($\mathcal{C}$) est la catégorie quotient

$$\text{diagr}(\mathcal{C}) = \text{DIAGR}(\mathcal{C})/\approx.$$

Lemme 2.9 *(La relation de connexion $\sim_{\overline{\delta}}$ est une relation d'équivalence)*
Soit un diagramme $\overline{\delta}$ sur $\mathcal{C}$, et A un objet de $\mathcal{C}$. La relation $\sim_{\overline{\delta}}$ est une relation d'équivalence sur l'ensemble de flèches de $\mathcal{C}$

$$\{u : A \to \delta(n) \ ; \ n \in \text{Nœud}(\delta^{\Phi})\}.$$

Preuve.

- La relation $\sim_{\overline{\delta}}$ est réflexive. En effet, pour toute flèche $u : A \to \delta(n)$ de $\mathcal{C}$,

$$u \sim_{\overline{\delta}} u \ [0_n].$$

- La relation $\sim_{\overline{\delta}}$ est symétrique. En effet, soit deux flèches $u : A \to \delta(n_0)$ et $v : A \to \delta(n_1)$ de $\mathcal{C}$. On a

$$u \sim_{\overline{\delta}} v \ [Z] \ \Rightarrow \ v \sim_{\overline{\delta}} u \ [\overline{Z}],$$

où $\overline{Z} : n_1 \rightsquigarrow n_0$ est le zigzag opposé de $Z : n_0 \rightsquigarrow n_1$ (cf. définition 2.4).

- La relation $\sim_{\overline{\delta}}$ est transitive. En effet, soit trois flèches $u : A \to \delta(n_0)$, $v : A \to \delta(n_1)$ et $w : A \to \delta(n_2)$ de $\mathcal{C}$. On a

$$u \sim_{\overline{\delta}} v \ [Z_1] \text{ et } v \sim_{\overline{\delta}} w \ [Z_2] \Rightarrow u \sim_{\overline{\delta}} w \ [Z],$$

où $Z : n_0 \rightsquigarrow n_2$ est le zigzag obtenu en mettant bout à bout $Z_1 : n_0 \rightsquigarrow n_1$ et $Z_2 : n_1 \rightsquigarrow n_2$.

□

Lemme 2.10 *($\sim_{\overline{\delta}}$ est une demi-congruence à droite)*
Soit $u : A \to \delta(n_0)$, $v : A \to \delta(n_k)$ et $w : A' \to A$. On a

$$u \sim_{\overline{\delta}} v \ [Z] \Rightarrow u \circ w \sim_{\overline{\delta}} v \circ w \ [Z]$$

(cf. figure 2.4).

Preuve. Application immédiate de la définition de $\sim_{\overline{\delta}}$. □

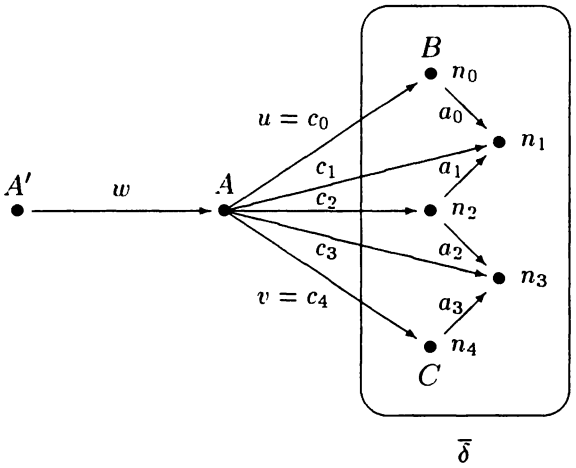


FIG. 2.4 –: $u \circ w \sim_{\overline{\delta}} v \circ w \ [Z]$

Lemme 2.11 *Soit $\overline{\alpha}$ et $\overline{\beta}$ deux diagrammes sur $\mathcal{C}$. Soit $\overline{\sigma} : \overline{\alpha} \to \overline{\beta}$ une flèche de* DIAGR($\mathcal{C}$). *Soit $u : A \to \alpha(m)$ et $v : A \to \alpha(n)$ deux flèches de $\mathcal{C}$. Alors,*

$$u \sim_{\overline{\alpha}} v \ [Z] \Rightarrow \sigma_m \circ u \sim_{\overline{\beta}} \sigma_n \circ v \ [\sigma^{\Phi}(Z)].$$

Preuve. Application de la définition de $\sim_{\overline{\beta}}$ et de $\overline{\sigma}$. □

Définition 2.31 (Congruence $\approx$)
Soit deux diagrammes $\overline{\alpha}$ et $\overline{\beta}$. Soit $\overline{\sigma}, \overline{\tau} : \overline{\alpha} \to \overline{\beta}$ deux flèches de DIAGR($\mathcal{C}$). On définit la relation $\approx$ sur les flèches de DIAGR($\mathcal{C}$) par

$$\overline{\sigma} \approx \overline{\tau} \Leftrightarrow \forall n \in \text{Nœud}(\alpha^\Phi),\ \sigma_n \sim_{\overline{\beta}} \tau_n.$$

Remarquons que c'est une bonne définition car $\forall n \in \text{Nœud}(\alpha^\Phi)$,

$$\sigma_n : \alpha(n) \to \beta(\sigma^\Phi(n)),$$
$$\tau_n : \alpha(n) \to \beta(\tau^\Phi(n)).$$

Proposition 2.1 *La relation $\approx$ est une congruence.*

Preuve.

- La relation $\approx$ est réflexive car $\sim_{\overline{\beta}}$ est réflexive. Soit $\overline{\sigma} : \overline{\alpha} \to \overline{\beta}$ une flèche de DIAGR($\mathcal{C}$).

$$\forall n \in \text{Nœud}(\alpha^\Phi),\ \sigma_n \sim_{\overline{\beta}} \sigma_n \Rightarrow \overline{\sigma} \approx \overline{\sigma}$$

- La relation $\approx$ est symétrique car $\sim_{\overline{\beta}}$ est symétrique. Soit $\overline{\sigma}, \overline{\tau} : \overline{\alpha} \to \overline{\beta}$ deux flèches de DIAGR($\mathcal{C}$).

$$\begin{array}{lll} \overline{\sigma} \approx \overline{\tau} & \Rightarrow \forall n \in \text{Nœud}(\alpha^\Phi),\ \sigma_n \sim_{\overline{\beta}} \tau_n & \text{(définition de } \approx) \\ & \Rightarrow \forall n \in \text{Nœud}(\alpha^\Phi),\ \tau_n \sim_{\overline{\beta}} \sigma_n & (\sim_{\overline{\beta}} \text{ symétrique}) \\ & \Rightarrow \overline{\tau} \approx \overline{\sigma} & \text{(définition de } \approx) \end{array}$$

- De même, la relation $\approx$ est transitive car $\sim_{\overline{\beta}}$ est transitive.

- Soit $\overline{\sigma}, \overline{\tau} : \overline{\alpha} \to \overline{\beta}$ et $\overline{\chi} : \overline{\delta} \to \overline{\alpha}$ trois flèches de DIAGR($\mathcal{C}$). Montrons que

$$\overline{\sigma} \approx \overline{\tau} \Rightarrow \overline{\sigma} \circ \overline{\chi} \approx \overline{\tau} \circ \overline{\chi}.$$

Supposons $\overline{\sigma} \approx \overline{\tau}$. Il suffit de montrer que

$$\forall n \in \text{Nœud}(\delta^\Phi),\ \sigma_{\chi^\Phi(n)} \circ \chi_n \sim_{\overline{\beta}} \tau_{\chi^\Phi(n)} \circ \chi_n.$$

$$\begin{array}{lll} \overline{\sigma} \approx \overline{\tau} & \Rightarrow \forall n \in \text{Nœud}(\delta^\Phi),\ \sigma_{\chi^\Phi(n)} \sim_{\overline{\beta}} \tau_{\chi^\Phi(n)} & \text{(définition de } \approx) \\ & \Rightarrow \forall n \in \text{Nœud}(\delta^\Phi),\ \sigma_{\chi^\Phi(n)} \circ \chi_n \sim_{\overline{\beta}} \tau_{\chi^\Phi(n)} \circ \chi_n & \text{(lemme 2.10)} \end{array}$$

- Soit $\overline{\sigma}, \overline{\tau} : \overline{\alpha} \to \overline{\beta}$ et $\overline{\xi} : \overline{\beta} \to \overline{\gamma}$ trois flèches de DIAGR($\mathcal{C}$). Montrons que

$$\overline{\sigma} \approx \overline{\tau} \Rightarrow \overline{\xi} \circ \overline{\sigma} \approx \overline{\xi} \circ \overline{\tau}.$$

Supposons que $\overline{\sigma} \approx \overline{\tau}$. Il suffit de montrer que

$$\forall n \in \text{Nœud}(\alpha^\Phi),\ \xi_{\sigma^\Phi(n)} \circ \sigma_n \sim_{\overline{\gamma}} \xi_{\tau^\Phi(n)} \circ \tau_n.$$

$$\begin{array}{lll} \overline{\sigma} \approx \overline{\tau} & \Rightarrow \forall n \in \text{Nœud}(\alpha^\Phi),\ \sigma_n \sim_{\overline{\beta}} \tau_n & \text{(définition de } \approx) \\ & \Rightarrow \forall n \in \text{Nœud}(\alpha^\Phi),\ \xi_{\sigma^\Phi(n)} \circ \sigma_n \sim_{\overline{\gamma}} \xi_{\tau^\Phi(n)} \circ \tau_n & \text{(lemme 2.11)} \end{array}$$

□

Lemme 2.12 *Soit deux catégories $\mathcal{C}$ et $\mathcal{D}$ et un foncteur $F : \mathcal{C} \to \mathcal{D}$. Soit deux flèches $\overline{\sigma}, \overline{\tau} : \overline{\alpha} \to \overline{\beta}$ de* DIAGR($\mathcal{C}$). *Alors,*

$$\overline{\sigma} \approx \overline{\tau} \Rightarrow \text{DIAGR}(F)(\overline{\sigma}) \approx \text{DIAGR}(F)(\overline{\tau}).$$

Preuve.

$$\begin{array}{lll} \overline{\sigma} \approx \overline{\tau} & & \\ \Rightarrow & \forall n \in \text{Nœud}(\alpha^\Phi),\ \sigma_n \sim_{\overline{\beta}} \tau_n & \text{(définition de } \approx) \\ \Rightarrow & \forall n \in \text{Nœud}(\alpha^\Phi),\ F(\sigma_n) \sim_{F \circ \overline{\beta}} F(\tau_n) & \text{(lemme 2.3)} \\ \Rightarrow & \text{DIAGR}(F)(\overline{\sigma}) \approx \text{DIAGR}(F)(\overline{\tau}) & \text{(définitions de DIAGR}(F) \text{ et } \approx) \end{array}$$

□

Définition 2.32 (Catégorie diagr($\mathcal{C}$))
Comme $\approx$ est une congruence (proposition 2.1), nous pouvons définir la catégorie diagr($\mathcal{C}$) comme le quotient de DIAGR($\mathcal{C}$) par $\approx$.

$$\text{diagr}(\mathcal{C}) = \text{DIAGR}(\mathcal{C})/\approx.$$

Les objets de diagr($\mathcal{C}$) sont les objets de DIAGR($\mathcal{C}$). Étant donné deux objets $\overline{\alpha}$ et $\overline{\beta}$ de diagr($\mathcal{C}$), une flèche entre $\overline{\alpha}$ et $\overline{\beta}$ est une classe d'équivalence modulo $\approx$ de flèches de DIAGR($\mathcal{C}$) entre $\overline{\alpha}$ et $\overline{\beta}$.

On a un foncteur de projection

$$\begin{array}{rccc} [-]_{\mathcal{C}} : & \text{DIAGR}(\mathcal{C}) & \to & \text{diagr}(\mathcal{C}) \\ & \overline{\alpha} & \mapsto & \overline{\alpha} \\ & \overline{\sigma} : \overline{\alpha} \to \overline{\beta} & \mapsto & [\overline{\sigma}] : \overline{\alpha} \to \overline{\beta} \end{array}$$

qui est l'identité sur les objets, et associe à toute flèche $\overline{\sigma} : \overline{\alpha} \to \overline{\beta}$ de DIAGR($\mathcal{C}$) sa classe d'équivalence $[\overline{\sigma}] : \overline{\alpha} \to \overline{\beta}$ modulo $\approx$ dans diagr($\mathcal{C}$).

Comme le foncteur de projection $[-]_{\mathcal{C}}$: DIAGR($\mathcal{C}$) $\to$ diagr($\mathcal{C}$) est l'identité sur les objets, on notera de la même façon un objet $\overline{\alpha}$ de DIAGR($\mathcal{C}$) et son image par $[-]_{\mathcal{C}}$ dans diagr($\mathcal{C}$).

Propriété 2.8 *Soit F : DIAGR($\mathcal{C}$) $\to \mathcal{D}$ un foncteur compatible avec $\approx$, c'est-à-dire tel que pour toutes flèches $\overline{\sigma}, \overline{\tau} : \overline{\alpha} \to \overline{\beta}$ de* DIAGR($\mathcal{C}$),

$$\overline{\sigma} \approx \overline{\tau} \;\Rightarrow\; F(\overline{\sigma}) = F(\overline{\tau}).$$

Alors, le foncteur F se factorise à travers $[-]_{\mathcal{C}}$. Autrement dit, il existe un unique foncteur G : diagr($\mathcal{C}$) $\to \mathcal{D}$ tel que $F = G \circ [-]_{\mathcal{C}}$.

Définition 2.33 (Foncteur diagr(F)) Soit deux catégories $\mathcal{C}$ et $\mathcal{D}$, et un foncteur $F : \mathcal{C} \to \mathcal{D}$. D'après le lemme 2.12, le foncteur $[-]_{\mathcal{D}} \circ$ DIAGR(F) : DIAGR($\mathcal{C}$) $\to$ diagr($\mathcal{D}$) est compatible avec $\approx$. Donc, d'après la propriété 2.8, il existe un unique foncteur

$$\text{diagr}(F) : \text{diagr}(\mathcal{C}) \to \text{diagr}(\mathcal{D})$$

tel que $\text{diagr}(F) \circ [-]_{\mathcal{C}} = [-]_{\mathcal{D}} \circ \text{DIAGR}(F)$.

Lemme 2.13 *L'application*

$$\begin{array}{rccc} \text{diagr} : & \mathbf{CAT} & \to & \mathbf{CAT} \\ & \mathcal{C} & \mapsto & \text{diagr}(\mathcal{C}) \\ & F : \mathcal{C} \to \mathcal{D} & \mapsto & \text{diagr}(F) : \text{diagr}(\mathcal{C}) \to \text{diagr}(\mathcal{D}) \end{array}$$

est un foncteur diagr : **CAT** $\to$ **CAT**.

Preuve. Application immédiate de la définition de diagr sur les flèches. □

Propriété 2.9
L'application $[-]$ qui à toute catégorie $\mathcal{C}$ associe le foncteur de projection

$$[-]_{\mathcal{C}} : \text{DIAGR}(\mathcal{C}) \to \text{diagr}(\mathcal{C})$$

est une transformation naturelle

$$[-] : \text{DIAGR} \dot{\to} \text{diagr}.$$

Preuve. Résulte immédiatement de la définition de diagr(F). □

Lemme 2.14 *Soit deux foncteurs G, G' : diagr($\mathcal{C}$) $\to \mathcal{D}$. Alors,*

$$G \circ [-]_{\mathcal{C}} \cong G' \circ [-]_{\mathcal{C}} \;\Rightarrow\; G \cong G'.$$

Preuve. Soit $\omega : G \circ [-]_{\mathcal{C}} \dot{\to} G' \circ [-]_{\mathcal{C}}$ l'isomorphisme naturel entre $G \circ [-]_{\mathcal{C}}$ et $G' \circ [-]_{\mathcal{C}}$. Comme $[-]_{\mathcal{C}} : \mathrm{DIAGR}(\mathcal{C}) \to \mathrm{diagr}(\mathcal{C})$ est l'identité sur les objets on peut définir une transformation naturelle $\xi : G \dot{\to} G'$ par

$$\xi_{\overline{\alpha}} = \omega_{\overline{\alpha}}.$$

Cela définit bien une transformation naturelle car pour toute flèche $\overline{\tau} : \overline{\alpha} \to \overline{\beta}$ de $\mathrm{diagr}(\mathcal{C})$, il existe une flèche $\overline{\sigma} : \overline{\alpha} \to \overline{\beta}$ de $\mathrm{diagr}(\mathcal{C})$ telle que $\overline{\tau} = [\overline{\sigma}]$. Comme ω est une transformation naturelle,

$$\begin{aligned} &\omega_{\overline{\beta}} \circ G([\overline{\sigma}]) = G'([\overline{\sigma}]) \circ \omega_{\overline{\alpha}} \\ &\quad \Rightarrow \quad \xi_{\overline{\beta}} \circ G(\overline{\tau}) = G'(\overline{\tau}) \circ \xi_{\overline{\alpha}} \end{aligned}$$

De plus, $\xi_{\overline{\alpha}}$ est un isomorphisme pour tout diagramme $\overline{\alpha}$ de $\mathrm{diagr}(\mathcal{C})$, donc on a bien un isomorphisme naturel

$$G \cong G'.$$

□

Jusqu'à la fin de ce paragraphe, nous considérons une catégorie $\mathcal{C}$ finiment cocomplète. Nous allons montrer que les colimites sont compatibles avec la relation de congruence $\approx$, c'est-à-dire que le foncteur $\mathrm{Colim}_{\mathcal{C}}$ envoie des flèches équivalentes sur des colimites égales. Intuitivement, cette propriété prouve que la relation de congruence n'est pas trop "forte", c'est-à-dire n'identifie pas trop de flèches, et permet de définir un foncteur

$$\mathrm{colim}_{\mathcal{C}} : \mathrm{diagr}(\mathcal{C}) \to \mathcal{C}.$$

Lemme 2.15 *Soit un diagramme* $\overline{\alpha}$ *de* $\mathrm{diagr}(\mathcal{C})$, B *un objet de* $\mathcal{C}$, *et deux flèches* $\overline{\sigma}, \overline{\tau} : \overline{\alpha} \to I_{\mathcal{C}}(B)$. *Alors,*

$$\overline{\sigma} \approx \overline{\tau} \;\Rightarrow\; \overline{\sigma} = \overline{\tau}.$$

Preuve. D'abord, $\sigma^{\Phi} = \tau^{\Phi}$ car $I_{\mathcal{C}}(B)$ est construit sur le graphe 1^{Φ}, qui ne comporte qu'un seul nœud $*$, et qu'un seul zigzag $0_* : * \rightsquigarrow *$. D'autre part, comme $\overline{\sigma} \approx \overline{\tau}$, pour tout nœud n de α^{Φ}, on a $\sigma_n \sim_{I_{\mathcal{C}}(B)} \tau_n$. Comme $0_* : * \rightsquigarrow *$ est le seul zigzag de 1^{Φ}, on a nécessairement $\sigma_n = \tau_n$. Par conséquent, $\overline{\sigma} = \overline{\tau}$. □

Proposition 2.2 *(Le foncteur* $\mathrm{Colim}_{\mathcal{C}}$ *est compatible avec* $\approx$*)*
Soit $\mathcal{C}$ *une catégorie finiment cocomplète. Le foncteur* $\mathrm{Colim}_{\mathcal{C}}$ *est compatible avec* $\approx$. *Autrement dit, soit* $\overline{\alpha}$ *et* $\overline{\beta}$ *deux diagrammes et* $\overline{\sigma}, \overline{\tau} : \overline{\alpha} \to \overline{\beta}$ *deux flèches de* $\mathrm{DIAGR}(\mathcal{C})$. *Alors,*

$$\overline{\sigma} \approx \overline{\tau} \Rightarrow \mathrm{Colim}_{\mathcal{C}}\, \overline{\sigma} = \mathrm{Colim}_{\mathcal{C}}\, \overline{\tau}.$$

Preuve. Soit $(A,\ \overline{\lambda} : \overline{\alpha} \to I_{\mathcal{C}}(A))$ et $(B,\ \overline{\lambda'} : \overline{\beta} \to I_{\mathcal{C}}(B))$ les cônes colimites des diagrammes $\overline{\alpha}$ et $\overline{\beta}$.

$$\begin{array}{lll} \overline{\sigma} \approx \overline{\tau} & & \\ \Rightarrow & \overline{\lambda'} \circ \overline{\tau} \approx \overline{\lambda'} \circ \overline{\sigma} & (\approx \text{ est une congruence}) \\ \Rightarrow & \overline{\lambda'} \circ \overline{\tau} = \overline{\lambda'} \circ \overline{\sigma} & (\text{lemme 2.15}) \\ \Rightarrow & I_{\mathcal{C}}(\mathrm{Colim}_{\mathcal{C}}\, \overline{\tau}) \circ \overline{\lambda} = \overline{\lambda'} \circ \overline{\sigma} & (\text{définition de } \mathrm{Colim}_{\mathcal{C}}\, \overline{\tau}) \\ \Rightarrow & \mathrm{Colim}_{\mathcal{C}}\, \overline{\sigma} = \mathrm{Colim}_{\mathcal{C}}\, \overline{\tau} & (\text{définition de } \mathrm{Colim}_{\mathcal{C}}\, \overline{\sigma}) \end{array}$$

□

Corollaire 2.1 *D'après la propriété* 2.8, *le foncteur* $\mathrm{Colim}_{\mathcal{C}} : \mathrm{DIAGR}(\mathcal{C}) \to \mathcal{C}$ *se factorise à travers* $[-]_{\mathcal{C}}$. *Autrement dit, il existe un unique foncteur*

$$\mathrm{colim}_{\mathcal{C}} : \mathrm{diagr}(\mathcal{C}) \to \mathcal{C}$$

tel que $\mathrm{colim}_{\mathcal{C}} \circ [-]_{\mathcal{C}} = \mathrm{Colim}_{\mathcal{C}}$.

Remarque 2.8 Dans la catégorie $\mathrm{diagr}(\mathcal{C})$, il existe plus d'isomorphismes entre diagrammes, et plus d'égalités entre flèches de diagrammes que dans la catégorie $\mathrm{DIAGR}(\mathcal{C})$. Néanmoins, dans le cas général,

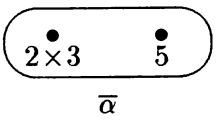
1. $\mathrm{colim}_{\mathcal{C}}\, \overline{\sigma} = \mathrm{colim}_{\mathcal{C}}\, \overline{\tau} \not\Rightarrow \overline{\sigma} = \overline{\tau}$;
2. $\mathrm{colim}_{\mathcal{C}}\, \overline{\alpha} \cong \mathrm{colim}_{\mathcal{C}}\, \overline{\beta} \not\Rightarrow \overline{\alpha} \cong \overline{\beta}$.

La raison intuitive est qu'on peut avoir des égalités entre flèches de $\mathcal{C}$ qui ne proviennent pas des propriétés générales des colimites, mais de la catégorie $\mathcal{C}$ elle-même.

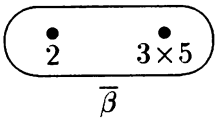
Contrexemple. On considère la catégorie $(\mathbb{N}, /)$ définie de la façon suivante.

- Les objets de $(\mathbb{N}, /)$ sont les entiers naturels.
- On a une flèche unique de n vers m si et seulement si n divise m.

$(\mathbb{N}, /)$ est une catégorie finiment cocomplète. (Par exemple, la somme catégorielle de m et n est le *ppcm* de m et n.)
Les diagrammes $\overline{\alpha}$ et $\overline{\beta}$ dessinés ci-dessous, ne sont pas isomorphes dans $\mathrm{diagr}(\mathbb{N}, /)$, et ont pourtant des colimites isomorphes.

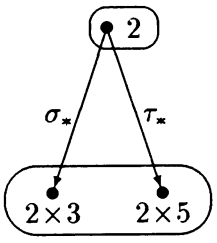


$\overline{\alpha}$



$\overline{\beta}$

Les deux diagrammes ont en effet pour colimite l'entier $2 \times 3 \times 5$.
De même, les flèches $\overline{\sigma}$ et $\overline{\tau}$, dessinées ci-dessous, ne sont pas égales dans $\mathrm{diagr}(\mathbb{N}, /)$, et ont pourtant des colimites égales : la flèche de l'entier 2 vers l'entier $2 \times 3 \times 5$.



Ici, la définition de la catégorie indique que deux flèches entre deux objets sont forcément égales. C'est la raison pour laquelle $\mathrm{colim}_{\mathcal{C}}\,\overline{\sigma} = \mathrm{colim}_{\mathcal{C}}\,\overline{\tau}$. Dans une autre catégorie, il n'y a *a priori* aucune raison pour que cette égalité soit vérifiée.

Théorème 2.5 *Soit $\mathcal{C}$ une catégorie finiment cocomplète.*

1. *Les deux foncteurs*
$$\mathrm{colim}_{\mathcal{C}} : \mathrm{diagr}(\mathcal{C}) \to \mathcal{C}$$
$$[-]_{\mathcal{C}} \circ I_{\mathcal{C}} : \mathcal{C} \to \mathrm{diagr}(\mathcal{C})$$
forment une adjonction
$$(\mathrm{colim}_{\mathcal{C}} \dashv [-]_{\mathcal{C}} \circ I_{\mathcal{C}}).$$

2. *La coünité de cette adjonction est la coünité de l'adjonction*
$$(\mathrm{Colim}_{\mathcal{C}} \dashv I_{\mathcal{C}}).$$

Preuve.

1. On définit une transformation naturelle $\overline{\eta'} : Id_{\mathrm{diagr}(\mathcal{C})} \dot{\to} [-]_{\mathcal{C}} \circ I_{\mathcal{C}} \circ \mathrm{colim}_{\mathcal{C}}$ de la façon suivante. Pour tout objet $\overline{\alpha}$ de $\mathrm{diagr}(\mathcal{C})$, on pose
$$\overline{\eta'}_{\overline{\alpha}} = [\overline{\eta}_{\overline{\alpha}}]$$
où $\overline{\eta} : Id_{\mathrm{DIAGR}(\mathcal{C})} \dot{\to} I_{\mathcal{C}} \circ \mathrm{Colim}_{\mathcal{C}}$ est l'unité de l'adjonction $(\mathrm{Colim}_{\mathcal{C}} \dashv I_{\mathcal{C}})$. C'est une bonne définition car le foncteur $[-]_{\mathcal{C}}$ est l'identité sur les objets. On vérifie facilement que $\overline{\eta'}$ est bien une transformation naturelle.

   Montrons maintenant que $(\mathrm{colim}_{\mathcal{C}} \dashv [-]_{\mathcal{C}} \circ I_{\mathcal{C}})$ est une adjonction dont l'unité est $\overline{\eta'}$. Soit un diagramme $\overline{\alpha}$ de $\mathrm{diagr}(\mathcal{C})$, B un objet de $\mathcal{C}$, et $[\overline{\sigma}] : \overline{\alpha} \to I_{\mathcal{C}}(B)$ une flèche de $\mathrm{diagr}(\mathcal{C})$. Il faut montrer qu'il existe une unique flèche de $\mathcal{C}$
$$\phi : \mathrm{colim}_{\mathcal{C}}\,\overline{\alpha} \to B$$
   telle que
$$[I_{\mathcal{C}}(\phi)] \circ [\overline{\eta}_{\overline{\alpha}}] = [\overline{\sigma}].$$

   Existence. Comme $(\mathrm{Colim}_{\mathcal{C}} \dashv I_{\mathcal{C}})$ est une adjonction dont l'unité est $\overline{\eta} : Id_{\mathrm{DIAGR}(\mathcal{C})} \dot{\to} I_{\mathcal{C}} \circ \mathrm{Colim}_{\mathcal{C}}$, il existe une unique flèche $\phi : \mathrm{Colim}_{\mathcal{C}}\,\overline{\alpha} \to B$ telle que
$$I_{\mathcal{C}}(\phi) \circ \overline{\eta}_{\overline{\alpha}} = \overline{\sigma}.$$
   En considérant $\overline{\alpha}$ comme un objet de $\mathrm{diagr}(\mathcal{C})$, la flèche ϕ a pour domaine $\mathrm{colim}_{\mathcal{C}}\,\overline{\alpha}$.

$$\begin{array}{ll} I_{\mathcal{C}}(\phi) \circ \overline{\eta}_{\overline{\alpha}} = \overline{\sigma} & \\ \Rightarrow\ [I_{\mathcal{C}}(\phi) \circ \overline{\eta}_{\overline{\alpha}}] = [\overline{\sigma}] & \\ \Rightarrow\ [I_{\mathcal{C}}(\phi)] \circ [\overline{\eta}_{\overline{\alpha}}] = [\overline{\sigma}] & ([-]_{\mathcal{C}} \text{ foncteur}) \\ \Rightarrow\ [I_{\mathcal{C}}(\phi)] \circ \overline{\eta'}_{\overline{\alpha}} = [\overline{\sigma}] & (\text{définition de } \overline{\eta'}) \end{array}$$

Unicité. Soit $\phi, \psi : \mathrm{colim}_{\mathcal{C}}\, \alpha \to B$ deux flèches de $\mathcal{C}$ telles que

$$[I_{\mathcal{C}}(\phi)] \circ \overline{\eta'}_{\overline{\alpha}} = [\overline{\sigma}],$$
$$[I_{\mathcal{C}}(\psi)] \circ \overline{\eta'}_{\overline{\alpha}} = [\overline{\sigma}].$$

On montre que $\phi = \psi$.

$$\begin{array}{lll}
[I_{\mathcal{C}}(\phi)] \circ \overline{\eta'}_{\overline{\alpha}} = [I_{\mathcal{C}}(\psi)] \circ \overline{\eta'}_{\overline{\alpha}} & & \\
\quad \Rightarrow \quad [I_{\mathcal{C}}(\phi)] \circ [\overline{\eta}_{\overline{\alpha}}] = [I_{\mathcal{C}}(\psi)] \circ [\overline{\eta}_{\overline{\alpha}}] & & \text{(définition de } \overline{\eta'}) \\
\quad \Rightarrow \quad [I_{\mathcal{C}}(\phi) \circ \overline{\eta}_{\overline{\alpha}}] = [I_{\mathcal{C}}(\psi) \circ \overline{\eta}_{\overline{\alpha}}] & & ([-]_{\mathcal{C}} \text{ foncteur}) \\
\quad \Rightarrow \quad I_{\mathcal{C}}(\phi) \circ \overline{\eta}_{\overline{\alpha}} \approx I_{\mathcal{C}}(\psi) \circ \overline{\eta}_{\overline{\alpha}} & & \text{(définition de } \approx) \\
\quad \Rightarrow \quad I_{\mathcal{C}}(\phi) \circ \overline{\eta}_{\overline{\alpha}} = I_{\mathcal{C}}(\psi) \circ \overline{\eta}_{\overline{\alpha}} & & \text{(lemme 2.15)} \\
\quad \Rightarrow \quad \phi = \psi & & \text{(adjonction } (\mathrm{Colim}_{\mathcal{C}} \dashv I_{\mathcal{C}}))
\end{array}$$

2. Considérons $\epsilon : \mathrm{Colim}_{\mathcal{C}} \circ I_{\mathcal{C}} \dot{\to} Id_{\mathcal{C}}$ la coünité de l'adjonction $(\mathrm{Colim}_{\mathcal{C}} \dashv I_{\mathcal{C}})$. Comme $\mathrm{Colim}_{\mathcal{C}} = \mathrm{colim}_{\mathcal{C}} \circ [-]_{\mathcal{C}}$, ϵ est bien une transformation naturelle

$$\epsilon : \mathrm{colim}_{\mathcal{C}} \circ [-]_{\mathcal{C}} \circ I_{\mathcal{C}} \dot{\to} Id_{\mathcal{C}}.$$

Pour montrer que ϵ est la coünité de l'adjonction $(\mathrm{colim}_{\mathcal{C}} \dashv [-]_{\mathcal{C}} \circ I_{\mathcal{C}})$: $\mathrm{diagr}(\mathcal{C}) \to \mathcal{C}$, on utilise le théorème 2.1.

Soit une flèche $g : \mathrm{colim}_{\mathcal{C}}\, \overline{\alpha} \to B$ de $\mathcal{C}$. Il faut montrer qu'il existe une unique flèche $[\overline{\sigma}] : \overline{\alpha} \to I_{\mathcal{C}}(B)$ de $\mathrm{diagr}(\mathcal{C})$ telle que

$$\epsilon_B \circ \mathrm{colim}_{\mathcal{C}}[\overline{\sigma}] = g.$$

Existence. Comme $(\mathrm{Colim}_{\mathcal{C}} \dashv I_{\mathcal{C}})$ est une adjonction, il existe une unique flèche $\overline{\sigma} : \overline{\alpha} \to I_{\mathcal{C}}(B)$ de $\mathrm{DIAGR}(\mathcal{C})$ telle que

$$\begin{array}{lll}
\epsilon_B \circ \mathrm{Colim}_{\mathcal{C}}\, \overline{\sigma} = g & & \\
\quad \Leftrightarrow \quad \epsilon_B \circ \mathrm{colim}_{\mathcal{C}}[\overline{\sigma}] = g & & \text{(définition de colim}_{\mathcal{C}}\text{)}
\end{array}$$

Unicité. Soit $[\overline{\sigma}], [\overline{\tau}] : \overline{\alpha} \to I_{\mathcal{C}}(B)$ deux flèches de $\mathrm{diagr}(\mathcal{C})$ telles que

$$\epsilon_B \circ \mathrm{colim}_{\mathcal{C}}[\overline{\sigma}] = g,$$
$$\epsilon_B \circ \mathrm{colim}_{\mathcal{C}}[\overline{\tau}] = g.$$

On montre que $[\overline{\sigma}] = [\overline{\tau}]$.

$$\begin{array}{lll}
\epsilon_B \circ \mathrm{colim}_{\mathcal{C}}[\overline{\sigma}] = \epsilon_B \circ \mathrm{colim}_{\mathcal{C}}[\overline{\tau}] & & \\
\quad \Rightarrow \quad \epsilon_B \circ \mathrm{Colim}_{\mathcal{C}}\, \overline{\sigma} = \epsilon_B \circ \mathrm{Colim}_{\mathcal{C}}\, \overline{\tau} & & \text{(définition de colim}_{\mathcal{C}}\text{)} \\
\quad \Rightarrow \quad \overline{\sigma} = \overline{\tau} & & \text{(adjonction } (\mathrm{Colim}_{\mathcal{C}} \dashv I_{\mathcal{C}})) \\
\quad \Rightarrow \quad [\overline{\sigma}] = [\overline{\tau}]. & &
\end{array}$$

□

2.3.8 La catégorie diagr($\mathcal{C}$) est finiment cocomplète

Dans ce paragraphe, nous considérons une catégorie $\mathcal{C}$ quelconque, et nous allons montrer que la catégorie diagr($\mathcal{C}$) est finiment cocomplète. Pour cela nous devons montrer que tout diagramme $\overline{\overline{\Gamma}}$ de DIAGR(diagr($\mathcal{C}$)) a une colimite dans diagr($\mathcal{C}$). Intuitivement, pour construire une colimite de $\overline{\overline{\Gamma}}$, on commence par construire un diagramme $\overline{\overline{\Delta}}$ de DIAGR$^2(\mathcal{C})$ construit sur le même graphe sous-jacent que $\overline{\overline{\Gamma}}$, tel que pour tout arc $A : N \to N'$, $\overline{\Delta(A)}$ est un représentant de la classe d'équivalence $\overline{\Gamma(A)}$ de flèches de DIAGR($\mathcal{C}$). Ensuite, l'aplatissement de $\overline{\overline{\Delta}}$ donne une colimite du diagramme $\overline{\overline{\Gamma}}$.

Lemme 2.16
Soit un objet $\overline{\overline{\Gamma}}$ *de* DIAGR(diagr($\mathcal{C}$)). *Il existe un objet* $\overline{\overline{\Delta}}$ *de* DIAGR$^2(\mathcal{C})$ *tel que*

$$\overline{\overline{\Gamma}} = \mathrm{DIAGR}([-]_{\mathcal{C}})(\overline{\overline{\Delta}}) = [-]_{\mathcal{C}} \circ \overline{\overline{\Delta}}.$$

Preuve. On définit un diagramme

$$\overline{\overline{\Delta}} = (\Delta^{\Phi},\ \Delta : \Delta^{\Phi} \to \mathrm{DIAGR}(\mathcal{C}))$$

de la façon suivante :

- Le graphe sous-jacent de $\overline{\overline{\Delta}}$ est le graphe sous-jacent de $\overline{\overline{\Gamma}}$. Autrement dit,

$$\Delta^{\Phi} = \Gamma^{\Phi}.$$

- Le foncteur $\Delta : \Delta^{\Phi} \to \mathrm{DIAGR}(\mathcal{C})$ est défini par son action sur les nœuds et sur les arcs du graphe Δ^{Φ}.
  - Tout nœud N de Δ^{Φ} est étiqueté par $\overline{\Gamma(N)}$.

$$\overline{\Delta(N)} = \overline{\Gamma(N)};$$

  - Pour tout arc $A : N \to N'$ de Δ^{Φ}, $\overline{\Gamma(A)} : \overline{\Gamma(N)} \to \overline{\Gamma(N')}$ est une flèche de diagr($\mathcal{C}$), c'est-à-dire une classe d'équivalence de flèches de DIAGR($\mathcal{C}$). On *choisit* un représentant $\overline{\Delta(A)}$ de cette classe d'équivalence. Autrement dit, $\overline{\Delta(A)}$ est tel que

$$\overline{\Gamma(A)} = [\overline{\Delta(A)}].$$

  Cela définit bien un foncteur car

$$\overline{\Delta(A)} : \overline{\Delta(N)} \to \overline{\Delta(N')}.$$

On vérifie immédiatement que

$$\overline{\overline{\Gamma}} = [-]_{\mathcal{C}} \circ \overline{\overline{\Delta}}.$$

□

Soit $\overline{\overline{\Gamma}}$ un objet de DIAGR(diagr($\mathcal{C}$)). D'après le lemme 2.16, il existe un objet $\overline{\overline{\Delta}}$ de DIAGR$^2(\mathcal{C})$ tel que $\overline{\overline{\Gamma}} = [-]_{\mathcal{C}} \circ \overline{\overline{\Delta}}$. Comme dans le paragraphe 2.3.6, on note

$$\overline{\delta} = \mathrm{Apl}_{\mathcal{C}}(\overline{\overline{\Delta}})$$

l'aplatissement de $\overline{\overline{\Delta}}$, et pour tout nœud N de Δ^{Φ}, on a une famille de flèches de DIAGR($\mathcal{C}$)

$$\overline{J_N} : \overline{\Delta(N)} \to \overline{\delta}.$$

On avait vu en remarque 2.7 que la famille de flèches

$$\overline{J_N} : \overline{\Delta(N)} \to \overline{\delta}$$

de DIAGR($\mathcal{C}$) ne permet pas de définir une flèche

$$\overline{\overline{J}} : \overline{\overline{\Delta}} \to I_{\mathrm{DIAGR}(\mathcal{C})}(\overline{\delta})$$

dans la catégorie DIAGR$^2(\mathcal{C})$. Par contre on a une flèche de $\overline{\overline{\Gamma}}$ vers $I_{\mathrm{diagr}(\mathcal{C})}(\overline{\delta})$ dans la catégorie DIAGR(diagr($\mathcal{C}$)), grâce au lemme suivant.

Lemme 2.17 *Pour tout arc $A : N \to N'$ de Δ^{Φ},*

$$\overline{J_{N'}} \circ \overline{\Delta(A)} \approx \overline{J_N}.$$

Preuve. Il suffit de montrer que

$$\forall n \in \mathrm{Nœud}(\Delta(N)^{\Phi}),\ (J_{N'})_{\Delta(A)^{\Phi}(n)} \circ \Delta(A)_n \sim_{\overline{\delta}} (J_N)_n.$$

Cette relation est vérifiée, car par définition de δ, on a

$$(J_{N'})_{\Delta(A)^{\Phi}(n)} \circ \Delta(A)_n = \delta(A, n) \circ (J_N)_n.$$

□

On définit une famille de flèches de diagr($\mathcal{C}$), en posant, pour tout nœud N de Δ^{Φ},

$$\overline{\eta_N} = [\overline{J_N}] : \overline{\Gamma(N)} \to \overline{\delta}.$$

Lemme 2.18 *$(\overline{\delta},\ \overline{\overline{\eta}} : \overline{\overline{\Gamma}} \to I_{\mathrm{diagr}(\mathcal{C})}(\overline{\delta}))$ est un cône sur $\overline{\overline{\Gamma}}$.*

Preuve. Il suffit de montrer que $\overline{\overline{\eta}} : \overline{\overline{\Gamma}} \to I_{\mathrm{diagr}(\mathcal{C})}(\overline{\delta})$ est une flèche de la catégorie DIAGR(diagr($\mathcal{C}$)), c'est-à-dire que pour tout arc $A : N \to N'$ de $\Delta(N)^{\Phi}$, on a

$$\overline{\eta_{N'}} \circ \overline{\Gamma(A)} = \overline{\eta_N}.$$

Pour tout arc $A : N \to N'$ de $\Gamma(N)^{\Phi}$, on a

$$\begin{array}{lll}
\overline{J_{N'}} \circ \overline{\Delta(A)} \approx \overline{J_N} & & \text{(lemme 2.17)} \\
\Rightarrow\ [\overline{J_{N'}} \circ \overline{\Delta(A)}] = [\overline{J_N}] & & \text{(définition de } [-]_{\mathcal{C}}) \\
\Rightarrow\ [\overline{J_{N'}}] \circ [\overline{\Delta(A)}] = [\overline{J_N}] & & ([-]_{\mathcal{C}} \text{ est un foncteur}) \\
\Rightarrow\ [\overline{J_{N'}}] \circ \overline{\Gamma(A)} = [\overline{J_N}] & & \text{(définition de } \overline{\Delta(A)}) \\
\Rightarrow\ \overline{\eta_{N'}} \circ \overline{\Gamma(A)} = \overline{\eta_N} & & \text{(définition de } \overline{\eta_N} \text{ et } \overline{\eta_{N'}})
\end{array}$$

Par conséquent, $(\overline{\delta},\ \overline{\overline{\eta}} : \overline{\overline{\Gamma}} \to I_{\mathrm{diagr}(\mathcal{C})}(\overline{\delta}))$ est un cône sur $\overline{\overline{\Gamma}}$. □

Lemme 2.19 *Soit un ensemble de flèches de* DIAGR($\mathcal{C}$)

$$\{\overline{K_N} : \overline{\Delta(N)} \to \overline{\alpha},\ \forall N \in \text{Nœud}(\Delta^\Phi)\}$$

tel que

$$\forall A : N \to N' \in \text{Arc}(\Delta^\Phi),\ \overline{K_{N'}} \circ \overline{\Delta(A)} \approx \overline{K_N}.$$

Alors,

1. *il existe une flèche* $\overline{\sigma} : \overline{\delta} \to \overline{\alpha}$ *de* DIAGR($\mathcal{C}$) *telle que*

$$\forall N \in \text{Nœud}(\Delta^\Phi),\ \overline{\sigma} \circ \overline{J_N} \approx \overline{K_N};$$

2. *pour toute flèche* $\overline{\tau} : \overline{\delta} \to \overline{\alpha}$ *de* DIAGR($\mathcal{C}$) *telle que*

$$\forall N \in \text{Nœud}(\Delta^\Phi),\ \overline{\tau} \circ \overline{J_N} \approx \overline{K_N},$$

on a

$$\overline{\sigma} \approx \overline{\tau}.$$

Preuve. Soit un ensemble de flèches de DIAGR($\mathcal{C}$)

$$\{\overline{K_N} : \overline{\Delta(N)} \to \overline{\alpha},\ \forall N \in \text{Nœud}(\Delta^\Phi)\}$$

tel que $\forall A : N \to N' \in \text{Arc}(\Delta^\Phi),\ \overline{K_{N'}} \circ \overline{\Delta(A)} \approx \overline{K_N}$.

1. Considérons la flèche de DIAGR($\mathcal{C}$)

$$\overline{\sigma} : \overline{\delta} \to \overline{\alpha} = (\sigma^\Phi : \delta^\Phi \rightsquigarrow \alpha^\Phi,\ \sigma : \delta \rightsquigarrow\!\!\!\!\cdot\ \alpha \circ \sigma^\Phi)$$

définie de la façon suivante.

- $\sigma^\Phi : \delta^\Phi \rightsquigarrow \alpha^\Phi$ est un morphisme généralisé de graphes.
  - $\forall (N, n) \in \text{Nœud}(\delta^\Phi),\ \sigma^\Phi(N, n) = K_N^\Phi(n)$.
  - $\forall (N, a) : (N, n) \to (N, n') \in \text{Arc}(\delta^\Phi)$,
    $\sigma^\Phi(N, a) = K_N^\Phi(a) : K_N^\Phi(n) \to K_N^\Phi(n')$.
  - $\forall (A, n) : (N, n) \to (N', \Delta(A)^\Phi(n)) \in \text{Arc}(\delta^\Phi),\ \sigma^\Phi(A, n) = Z$,
    où $Z : K_N^\Phi(n) \rightsquigarrow K_{N'}^\Phi(\Delta(A)(n))$ est le zigzag tel que
    $$(K_{N'})_{\Delta(A)^\Phi(n)} \circ \Delta(A)_n \sim_{\overline{\alpha}} (K_N)_n\ [Z].$$
- $\sigma : \delta \rightsquigarrow\!\!\!\!\cdot\ \alpha \circ \sigma^\Phi$ est la transformation naturelle généralisée définie par
  $$\forall (N, n) \in \text{Nœud}(\delta^\Phi),\ \sigma_{(N,n)} = (K_N)_n \circ (J_N)_n^{-1}.$$

Pour prouver que cela définit bien une flèche de DIAGR($\mathcal{C}$), il faut montrer que $\sigma : \delta \rightsquigarrow\!\!\!\!\cdot\ \alpha \circ \sigma^\Phi$ est bien une transformation naturelle généralisée c'est-à-dire que

(a) $\forall (N, a) : (N, n) \to (N, n') \in \text{Arc}(\delta^\Phi)$,
$$\sigma_{(N,n')} \circ \delta(N, a) \sim_{\overline{\alpha}} \sigma_{(N,n)}\ [\sigma^\Phi(N, a)]$$

(b) $\forall (A,n) : (N,n) \to (N', \Delta(A)^\Phi(n)) \in \mathrm{Arc}(\delta^\Phi)$,
$$\sigma_{(N',\Delta(A)^\Phi(n))} \circ \delta(A,n) \sim_{\overline{\alpha}} \sigma_{(N,n)} \ [\sigma^\Phi(A,n)]$$

(a) Comme K_N est une flèche de DIAGR($\mathcal{C}$),

$(K_N)_{n'} \circ \Delta(N)(a) \sim_{\overline{\alpha}} (K_N)_n \ [\sigma^\Phi(N,a)]$
$\Rightarrow$ $(K_N)_{n'} \circ \Delta(N)(a) \circ (J_N)_n^{-1} \sim_{\overline{\alpha}} (K_N)_n \circ (J_N)_n^{-1}$ (lemme 2.10)
$\Rightarrow$ $(K_N)_{n'} \circ (J_N)_{n'} \circ \delta(N,a) \sim_{\overline{\alpha}} (K_N)_n \circ (J_N)_n^{-1}$ (définition de $\delta(N,a)$)
$\Rightarrow$ $\sigma_{(N,n')} \circ \delta(N,a) \sim_{\overline{\alpha}} \sigma_{(N,n)} \ [\sigma^\Phi(N,a)]$ (définition de $\sigma_{(N,n)}$)

(b) Par définition de $\sigma(A,n)^\Phi$,

$(K_{N'})_{\Delta(A)^\Phi(n)} \circ \Delta(A)_n \sim_{\overline{\alpha}} (K_N)_n \ [\sigma^\Phi(A,n)]$
$\Rightarrow$ $(K_{N'})_{\Delta(A)^\Phi(n)} \circ \Delta(A)_n \circ (J_N)_n^{-1} \sim_{\overline{\alpha}} (K_N)_n \circ (J_N)_n^{-1} \ [\sigma^\Phi(A,n)]$
$\Rightarrow$ $(K_{N'})_{\Delta(A)^\Phi(n)} \circ (J_{N'})^{-1}_{\Delta(A)^\Phi(n)} \circ \delta(A,n) \sim_{\overline{\alpha}} (K_N)_n \circ (J_N)_n^{-1} \ [\sigma^\Phi(A,n)]$
$\Rightarrow$ $\sigma_{(N',\Delta(A)^\Phi(n))} \circ \delta(A,n) \sim_{\overline{\alpha}} \sigma_{(N,n)} \ [\sigma(A,n)^\Phi]$

Il reste à montrer que $\forall N \in \mathrm{Nœud}(\Delta^\Phi)$, $\overline{\sigma} \circ \overline{J_N} \approx \overline{K_N}$. Par définition de $\overline{\sigma}$,

$\forall (N,n) \in \mathrm{Nœud}(\delta^\Phi)$, $\sigma_{(N,n)} = (K_N)_n \circ (J_N)_n^{-1}$
$\Rightarrow$ $\forall (N,n) \in \mathrm{Nœud}(\delta^\Phi)$, $\sigma_{(N,n)} \circ (J_N)_n = (K_N)_n$
$\Rightarrow$ $\forall N \in \mathrm{Nœud}(\Delta^\Phi)$, $\forall n \in \mathrm{Nœud}(\Delta(N)^\Phi)$, $\sigma_{(N,n)} \circ (J_N)_n \sim_{\overline{\alpha}} (K_N)_n$
$\Rightarrow$ $\forall N \in \mathrm{Nœud}(\Delta^\Phi)$, $\overline{\sigma} \circ \overline{J_N} \approx \overline{K_N}$

2. Soit $\overline{\tau} : \overline{\delta} \to \overline{\alpha}$ une flèche de DIAGR($\mathcal{C}$) telle que
$$\forall N \in \mathrm{Nœud}(\Delta^\Phi), \ \overline{\tau} \circ \overline{J_N} \approx \overline{K_N}.$$

On montre que $\overline{\sigma} \approx \overline{\tau}$.

$\forall N \in \mathrm{Nœud}(\Delta^\Phi)$, $\overline{\tau} \circ \overline{J_N} \approx \overline{K_N}$
$\Rightarrow$ $\forall N$, $\forall n \in \mathrm{Nœud}(\Delta(N)^\Phi)$, $\tau_{(N,n)} \circ (J_N)_n \sim_{\overline{\alpha}} (K_N)_n$
$\Rightarrow$ $\forall (N,n) \in \mathrm{Nœud}(\delta^\Phi)$, $\tau_{(N,n)} \circ (J_N)_n \sim_{\overline{\alpha}} (K_N)_n$
$\Rightarrow$ $\forall (N,n) \in \mathrm{Nœud}(\delta^\Phi)$, $\tau_{(N,n)} \sim_{\overline{\alpha}} (K_N)_n \circ (J_N)_n^{-1}$
$\Rightarrow$ $\forall (N,n) \in \mathrm{Nœud}(\delta^\Phi)$, $\tau_{(N,n)} \sim_{\overline{\alpha}} \sigma_{(N,n)}$
$\Rightarrow$ $\overline{\tau} \approx \overline{\sigma}$

□

Lemme 2.20 *Le cône* $(\overline{\delta}, \ \overline{\overline{\eta}} : \overline{\overline{\Gamma}} \to I_{\mathrm{diagr}(\mathcal{C})}(\overline{\delta}))$ *est un cône colimite de* $\overline{\overline{\Gamma}}$.

Preuve. Soit $(\overline{\alpha}, \ \overline{\overline{\lambda}} : \overline{\overline{\Gamma}} \to I_{\mathrm{diagr}(\mathcal{C})}(\overline{\alpha}))$ un autre cône sur $\overline{\overline{\Gamma}}$. Nous devons montrer qu'il existe une unique flèche
$$[\overline{\sigma}] : \overline{\delta} \to \overline{\alpha}$$
de diagr($\mathcal{C}$) telle que
$$I_{\mathrm{diagr}(\mathcal{C})}([\overline{\sigma}]) \circ \overline{\overline{\eta}} = \overline{\overline{\lambda}}.$$

Existence. Pour tout nœud N de Γ^Φ, $\overline{\lambda_N} : \overline{\Gamma(N)} \to \overline{\alpha}$ est une flèche de diagr($\mathcal{C}$). On choisit un représentant $\overline{K_N} : \overline{\Gamma(N)} \to \overline{\alpha}$ dans DIAGR($\mathcal{C}$) de $\overline{\lambda_N}$. Autrement dit, on a

$$\forall N \in \text{Nœud}(\Gamma^\Phi),\ [\overline{K_N}] = \overline{\lambda_N}.$$

Comme $\overline{\overline{\lambda}} : \overline{\overline{\Gamma}} \to I_{\text{diagr}(\mathcal{C})}(\overline{\alpha})$ est une flèche de DIAGR(diagr($\mathcal{C}$)), pour tout arc $A : N \to N'$ du graphe $\Gamma^\Phi = \Delta^\Phi$, on a

$$\begin{array}{ll}
\overline{\lambda_{N'}} \circ \overline{\Gamma(A)} = \overline{\lambda_N} & \\
\quad \Rightarrow\ [\overline{K_{N'}}] \circ [\overline{\Delta(A)}] = [\overline{K_N}] & \text{(définition de } \overline{K_N}, \overline{K_{N'}} \text{ et } \overline{\overline{\Delta}}) \\
\quad \Rightarrow\ [\overline{K_{N'}} \circ \overline{\Delta(A)}] = [\overline{K_N}] & ([-]_{\mathcal{C}} \text{ foncteur}) \\
\quad \Rightarrow\ \overline{K_{N'}} \circ \overline{\Delta(A)} \approx \overline{K_N} & \text{(définition de diagr}(\mathcal{C}))
\end{array}$$

donc, d'après le lemme 2.19 (1), il existe une flèche $\overline{\sigma} : \overline{\delta} \to \overline{\alpha}$ de DIAGR($\mathcal{C}$) telle que

$$\forall N \in \text{Nœud}(\Delta^\Phi),\ \overline{\sigma} \circ \overline{J_N} \approx \overline{K_N}.$$

$$\begin{array}{l}
\forall N \in \text{Nœud}(\Gamma^\Phi),\ \overline{\sigma} \circ \overline{J_N} \approx \overline{K_N} \\
\quad \Rightarrow\ \forall N \in \text{Nœud}(\Gamma^\Phi),\ [\overline{\sigma} \circ \overline{J_N}] = [\overline{K_N}] \\
\quad \Rightarrow\ \forall N \in \text{Nœud}(\Gamma^\Phi),\ [\overline{\sigma}] \circ [\overline{J_N}] = [\overline{K_N}] \\
\quad \Rightarrow\ \forall N \in \text{Nœud}(\Gamma^\Phi),\ [\overline{\sigma}] \circ \overline{\eta_N} = \overline{\lambda_N} \\
\quad \Rightarrow\ I_{\text{diagr}(\mathcal{C})}([\overline{\sigma}]) \circ \overline{\overline{\eta}} = \overline{\overline{\lambda}}
\end{array}$$

Unicité. Soit $[\overline{\tau}] : \overline{\delta} \to \overline{\alpha}$ une flèche de diagr($\mathcal{C}$) telle que

$$I_{\text{diagr}(\mathcal{C})}([\overline{\tau}]) \circ \overline{\overline{\eta}} = \overline{\overline{\lambda}}.$$

$$\begin{array}{l}
I_{\text{diagr}(\mathcal{C})}([\overline{\sigma}]) \circ \overline{\overline{\eta}} = \overline{\overline{\lambda}} \\
\quad \Rightarrow\ \forall N \in \text{Nœud}(\Gamma^\Phi),\ [\overline{\tau}] \circ \overline{\eta_N} = \overline{\lambda_N} \\
\quad \Rightarrow\ \forall N \in \text{Nœud}(\Gamma^\Phi),\ [\overline{\tau}] \circ [\overline{J_N}] = [\overline{K_N}] \\
\quad \Rightarrow\ \forall N \in \text{Nœud}(\Gamma^\Phi),\ [\overline{\tau} \circ \overline{J_N}] = [\overline{K_N}] \\
\quad \Rightarrow\ \forall N \in \text{Nœud}(\Gamma^\Phi),\ \overline{\tau} \circ \overline{J_N} \approx \overline{K_N}
\end{array}$$

donc d'après le lemme 2.19 (2), $\sigma \approx \tau$, et par conséquent $[\sigma] = [\tau]$.

□

Théorème 2.6 *La catégorie* diagr($\mathcal{C}$) *est finiment cocomplète.*

Preuve. Conséquence immédiate du lemme 2.20. □

Remarque 2.9 Dans cette preuve, nous avons montré l'existence d'une colimite pour le diagramme $\overline{\overline{\Gamma}}$. Pour chaque arc de Γ^Φ, nous avons effectué un choix de représentant $\overline{\Delta(A)}$ pour la flèche $\overline{\Gamma(A)}$. Un choix différent de représentant conduit à un autre diagramme colimite (isomorphe) pour $\overline{\overline{\Gamma}}$. Nous avons donc défini la colimite de $\overline{\overline{\Gamma}}$ *à un isomorphisme près.*

Une colimite est définie à un isomorphisme près, mais lorsque les classes d'équivalence $\overline{\Gamma(A)}$ ne contiennent qu'un élément, alors il n'existe qu'un seul diagramme $\overline{\overline{\Delta}}$ tel que $\overline{\overline{\Gamma}} = [-]_{\mathcal{C}} \circ \overline{\overline{\Delta}}$. Soit un diagramme $\overline{\alpha}$ de DIAGR($\mathcal{C}$). Considérons le diagramme de DIAGR(diagr($\mathcal{C}$))

$$\overline{\overline{\Gamma}} = \mathrm{DIAGR}([-]_{\mathcal{C}} \circ I_{\mathcal{C}})(\overline{\alpha}) = [-]_{\mathcal{C}} \circ I_{\mathcal{C}} \circ \overline{\alpha}.$$

Pour tout arc $a : n \to n'$ du graphe $\Gamma^{\Phi} = \alpha^{\Phi}$,

$$\Gamma(a) = [I_{\mathcal{C}}(\alpha(a))],$$

et le seul élément de la classe $\Gamma(a)$ est $I_{\mathcal{C}}(\alpha(a))$. Donc

$$\overline{\overline{\Delta}} = I_{\mathcal{C}} \circ \overline{\alpha} = \mathrm{DIAGR}(I_{\mathcal{C}})(\overline{\alpha}).$$

D'après le lemme 2.8,

$$\mathrm{Apl}_{\mathcal{C}}(\overline{\overline{\Delta}}) = \mathrm{Apl}_{\mathcal{C}}(\mathrm{DIAGR}(I_{\mathcal{C}})(\overline{\alpha})) = \overline{\alpha}.$$

Finalement, la colimite de $\overline{\overline{\Gamma}}$ est donc $\overline{\alpha}$.

Lemme 2.21 *Soit une catégorie $\mathcal{C}$.*

$$\mathrm{Colim}_{\mathrm{diagr}(\mathcal{C})} \circ \mathrm{DIAGR}([-]_{\mathcal{C}} \circ I_{\mathcal{C}}) = [-]_{\mathcal{C}}.$$

Preuve.

- Pour tout diagramme $\overline{\alpha}$ de DIAGR($\mathcal{C}$),

$$\mathrm{Colim}_{\mathrm{diagr}(\mathcal{C})}(\mathrm{DIAGR}([-]_{\mathcal{C}} \circ I_{\mathcal{C}})(\overline{\alpha})) = \overline{\alpha}.$$

- Soit une flèche $\overline{\sigma} : \overline{\alpha} \to \overline{\beta}$ de DIAGR($\mathcal{C}$). Soit

$$(\overline{\alpha},\ \overline{\overline{\eta}} : [-]_{\mathcal{C}} \circ I_{\mathcal{C}} \circ \overline{\alpha} \to I_{\mathrm{diagr}(\mathcal{C})}(\overline{\alpha})$$

et

$$(\overline{\beta},\ \overline{\overline{\eta'}} : [-]_{\mathcal{C}} \circ I_{\mathcal{C}} \circ \overline{\beta} \to I_{\mathrm{diagr}(\mathcal{C})}(\overline{\beta})$$

les cône colimites respectifs des diagrammes $[-]_{\mathcal{C}} \circ I_{\mathcal{C}} \circ \overline{\alpha}$ et $[-]_{\mathcal{C}} \circ I_{\mathcal{C}} \circ \overline{\beta}$.

Posons

$$\overline{\overline{\Sigma}} = [-]_{\mathcal{C}} \circ I_{\mathcal{C}} \circ \overline{\sigma}.$$

Il faut montrer que

$$\mathrm{Colim}_{\mathrm{diagr}(\mathcal{C})} \overline{\overline{\Sigma}} = [\overline{\sigma}].$$

Comme $\mathrm{Colim}_{\mathrm{diagr}(\mathcal{C})} \overline{\overline{\Sigma}}$ est l'unique flèche telle que

$$I_{\mathrm{diagr}(\mathcal{C})}(\mathrm{Colim}_{\mathrm{diagr}(\mathcal{C})} \overline{\overline{\Sigma}}) \circ \overline{\overline{\eta}} = \overline{\overline{\eta'}} \circ \overline{\overline{\Sigma}},$$

il suffit de prouver

$$I_{\mathrm{diagr}(\mathcal{C})}([\overline{\sigma}]) \circ \overline{\overline{\eta}} = \overline{\overline{\eta'}} \circ \overline{\overline{\Sigma}}$$
$$\Leftrightarrow \quad \forall n \in \mathrm{Nœud}(\alpha^{\Phi}),\ [\overline{\sigma}] \circ \overline{\eta_n} = \overline{\eta'}_{\sigma^{\Phi}(n)} \circ \overline{\Sigma_n}.$$

□

Remarque 2.10 Ce lemme montre que le foncteur $[-]_{\mathcal{C}} \circ I_{\mathcal{C}} : \mathcal{C} \to \text{diagr}(\mathcal{C})$ est dense, c'est-à-dire que tout diagramme $\overline{\alpha}$ est colimite d'un diagramme construit sur les images des objets et flèches de $\mathcal{C}$ dans diagr($\mathcal{C}$). En effet, tout diagramme $\overline{\alpha}$ de diagr($\mathcal{C}$) est la colimite du diagramme $\overline{\overline{\Gamma}} = \text{DIAGR}([-]_{\mathcal{C}} \circ I_{\mathcal{C}})(\overline{\alpha})$.

2.3.9 La catégorie diagr($\mathcal{C}$) est une complétion de $\mathcal{C}$ par colimites finies

La catégorie diagr($\mathcal{C}$) a une propriété importante: c'est une complétion de $\mathcal{C}$ par colimites finies. Intuitivement, cela signifie que diagr($\mathcal{C}$) est une catégorie "minimum" qui contient $\mathcal{C}$, et toutes les colimites finies que l'on peut construire à partir d'objets et de flèches de $\mathcal{C}$.

Soit une catégorie $\mathcal{C}$ quelconque, et $\mathcal{D}$ une catégorie finiment cocomplète. Soit un foncteur $F : \mathcal{C} \to \mathcal{D}$. On considère le foncteur

$$G = \text{Colim}_{\mathcal{D}} \circ \text{DIAGR}(F) : \text{DIAGR}(\mathcal{C}) \to \mathcal{D}.$$

Lemme 2.22 *On a un isomorphisme naturel* $G \circ I_{\mathcal{C}} \cong F$.

Preuve. D'après le théorème 2.4, $\epsilon : \text{Colim}_{\mathcal{D}} \circ I_{\mathcal{D}} \dot{\to} Id_{\mathcal{D}}$, coünité de l'adjonction $(\text{Colim}_{\mathcal{D}} \dashv I_{\mathcal{D}})$ est un isomorphisme naturel. Par conséquent,

$$\begin{array}{lll} \text{Colim}_{\mathcal{D}} \circ I_{\mathcal{D}} \cong Id_{\mathcal{D}} & & \\ \Rightarrow \ \text{Colim}_{\mathcal{D}} \circ I_{\mathcal{D}} \circ F \cong F & & \text{(composition par } F\text{)} \\ \Rightarrow \ \text{Colim}_{\mathcal{D}} \circ \text{DIAGR}(F) \circ I_{\mathcal{C}} \cong F & & (I : Id_{\mathbf{CAT}} \dot{\to} \text{DIAGR}) \\ \Rightarrow \ G \circ I_{\mathcal{C}} \cong F & & \text{(définition de } G\text{)} \end{array}$$

□

Soit $\overline{\overline{\Delta}}$ un objet de $\text{DIAGR}^2(\mathcal{C})$, $\overline{\delta} = \text{Apl}_{\mathcal{C}}(\overline{\overline{\Delta}})$ l'aplatissement de $\overline{\overline{\Delta}}$, et la famille de flèches de DIAGR($\mathcal{C}$)

$$\overline{J_N} : \overline{\Delta(N)} \to \overline{\delta}, \ \forall N \in \text{Nœud}(\Delta^{\Phi}).$$

Pour tout nœud N de Δ^{Φ}, on considère la flèche

$$\lambda_N = G(\overline{J_N}) : G(\overline{\Delta(N)}) \to G(\overline{\delta})$$

dans $\mathcal{D}$.
Nous allons considérer les diagrammes suivants.

- Le diagramme

$$F \circ \overline{\delta} = (\delta^{\Phi}, \ F \circ \delta : \delta^{\Phi} \to \mathcal{D})$$

  est un objet de DIAGR($\mathcal{D}$), qui a pour cône colimite

$$(G(\overline{\delta}), \ \overline{\eta}_{F \circ \overline{\delta}} : F \circ \overline{\delta} \to I_{\mathcal{D}}(G(\overline{\delta}))).$$

- Pour tout nœud N de Δ^Φ, on a un diagramme

$$F \circ \overline{\Delta(N)} = (\Delta(N)^\Phi,\ F \circ \Delta(N) : \Delta(N)^\Phi \to \mathcal{D})$$

dans DIAGR($\mathcal{D}$), qui a pour cône colimite

$$(G(\overline{\Delta(N)}),\ \overline{\eta}_{F \circ \overline{\Delta(N)}} : F \circ \overline{\Delta(N)} \to I_{\mathcal{D}}(G(\overline{\Delta(N)}))).$$

Par définition, $\lambda_N = G(\overline{J_N}) = \mathrm{Colim}_{\mathcal{D}}(\mathrm{DIAGR}(F)(\overline{J_N}))$, donc λ_N est l'unique flèche telle que

$$I_{\mathcal{D}}(\lambda_N) \circ \overline{\eta}_{F \circ \overline{\Delta(N)}} = \overline{\eta}_{F \circ \overline{\delta}} \circ \mathrm{DIAGR}(F)(\overline{J_N}). \tag{1}$$

Lemme 2.23 *$\overline{\lambda} : G \circ \overline{\overline{\Delta}} \to I_{\mathcal{D}}(G(\overline{\delta}))$ est une flèche de* DIAGR($\mathcal{D}$).

Preuve. Il suffit de montrer que pour tout arc $A : N \to N'$ de Δ^Φ, on a

$$\lambda_{N'} \circ G(\overline{\Delta(A)}) = \lambda_N.$$

$\overline{J_{N'}} \circ \overline{\Delta(A)} \approx \overline{J_N}$		(lemme 2.17)
$\Rightarrow$	$\mathrm{DIAGR}(F)(\overline{J_{N'}} \circ \overline{\Delta(A)}) \approx \mathrm{DIAGR}(F)(\overline{J_N})$	(lemme 2.12)
$\Rightarrow$	$\mathrm{Colim}_{\mathcal{D}}(\mathrm{DIAGR}(F)(\overline{J_{N'}} \circ \overline{\Delta(A)}))$	
	$= \mathrm{Colim}_{\mathcal{D}}(\mathrm{DIAGR}(F)(\overline{J_N}))$	(proposition 2.2)
$\Rightarrow$	$G(\overline{J_{N'}} \circ \overline{\Delta(A)}) = G(\overline{J_N})$	(définition de G)
$\Rightarrow$	$G(\overline{J_{N'}}) \circ G(\overline{\Delta(A)}) = G(\overline{J_N})$	(G foncteur)
$\Rightarrow$	$\lambda_{N'} \circ G(\overline{\Delta(A)}) = \lambda_N$	(définition de λ_N et $\lambda_{N'}$)

□

Lemme 2.24 *$(G(\overline{\delta}),\ \overline{\lambda} : G \circ \overline{\overline{\Delta}} \to I_{\mathcal{D}}(G(\overline{\delta})))$ est un cône colimite du diagramme $G \circ \overline{\overline{\Delta}}$ de* DIAGR($\mathcal{D}$).

Preuve. Soit un cône $(A,\ \overline{\mu} : G \circ \overline{\overline{\Delta}} \to I_{\mathcal{D}}(A))$ sur le diagramme $G \circ \overline{\overline{\Delta}}$. $\overline{\mu}$ est donc une flèche de DIAGR($\mathcal{D}$). Nous allons montrer qu'il existe une unique flèche

$$\phi : G(\overline{\delta}) \to A$$

de $\mathcal{D}$ telle que

$$I_{\mathcal{D}}(\phi) \circ \overline{\lambda} = \overline{\mu}.$$

Existence. Pour tout nœud N de Δ^Φ et tout nœud n de $\Delta(N)^\Phi$, on pose

$$\psi_{N,n} = \mu_N \circ (\eta_{F\circ\overline{\Delta(N)}})_n \circ F((J_N)_n^{-1}) : F(\delta(N,n)) \to A.$$

Cette définition est correcte car

$$\begin{aligned} &F((J_N)_n^{-1}) : F(\delta(N,n)) \to F(\Delta(N)(n)) \\ &(\eta_{F\circ\overline{\Delta(N)}})_n : F(\Delta(N)(n)) \to G(\overline{\Delta(N)}) \\ &\mu_N : G(\overline{\Delta(N)}) \to A. \end{aligned}$$

On montre que

$$\overline{\psi} : F \circ \overline{\delta} \to I_{\mathcal{D}}(A)$$

est une flèche de DIAGR($\mathcal{D}$).

- Pour tout arc $(N,a) : (N,m) \to (N,n)$ de δ^Φ,

$$\begin{aligned} &\psi_{N,n} \circ F(\delta(N,a)) \\ &\quad = \mu_N \circ (\eta_{F\circ\Delta(N)})_n \circ F((J_N)_n^{-1}) \circ F(\delta(N,a)) && \text{(définition de } \psi_{N,n}) \\ &\quad = \mu_N \circ (\eta_{F\circ\Delta(N)})_n \circ F((J_N)_n^{-1} \circ \delta(N,a)) && (F \text{ foncteur}) \\ &\quad = \mu_N \circ (\eta_{F\circ\Delta(N)})_n \circ F(\Delta(N)(a) \circ (J_N)_m^{-1}) && \text{(définition de } \delta(N,a)) \\ &\quad = \mu_N \circ (\eta_{F\circ\Delta(N)})_n \circ F(\Delta(N)(a)) \circ F((J_N)_m^{-1}) && (F \text{ foncteur}) \\ &\quad = \mu_N \circ (\eta_{F\circ\Delta(N)})_m \circ F((J_N)_m^{-1}) && (\overline{\eta}_{F\circ\overline{\Delta(N)}} \text{ cône}) \\ &\quad = \psi_{N,m} && \text{(définition de } \psi_{M,m}) \end{aligned}$$

- Pour tout arc $(A,m) : (M,m) \to (N,n)$ de δ^Φ, où $A : M \to N \in \mathrm{Arc}(\Delta^\Phi)$, $m \in \mathrm{Nœud}(\Delta(M)^\Phi)$, et $n = \Delta(A)^\Phi(m) \in \mathrm{Nœud}(\Delta(N)^\Phi)$,

$$\begin{aligned} &\psi_{N,n} \circ F(\delta(A,m)) \\ &\quad = \mu_N \circ (\eta_{F\circ\Delta(N)})_n \circ F((J_N)_n^{-1}) \circ F(\delta(A,m)) && \text{(définition de } \psi_{N,n}) \\ &\quad = \mu_N \circ (\eta_{F\circ\Delta(N)})_n \circ F((J_N)_n^{-1} \circ \delta(A,m)) && (F \text{ foncteur}) \\ &\quad = \mu_N \circ (\eta_{F\circ\Delta(N)})_n \circ F(\Delta(A)_m) \circ F((J_M)_m^{-1}) && \text{(définition de } \delta(A,m)) \\ &\quad = \mu_N \circ G(\overline{\Delta(A)}) \circ (\eta_{F\circ\Delta(M)})_m \circ F((J_M)_m^{-1}) && \text{(définition de } G) \\ &\quad = \mu_M \circ (\eta_{F\circ\Delta(M)})_m \circ F((J_M)_m^{-1}) && (\overline{\mu} \text{ cône}) \\ &\quad = \psi_{M,m} && \text{(définition de } \psi_{M,m}) \end{aligned}$$

On a donc bien une flèche

$$\overline{\psi} : F \circ \overline{\delta} \to I_{\mathcal{D}}(A)$$

dans $\mathcal{D}$. De plus on vérifie immédiatemment que

$$\overline{\psi} \circ \mathrm{DIAGR}(F)(\overline{J_N}) = I_{\mathcal{D}}(\mu_N) \circ \overline{\eta}_{F\circ\overline{\Delta(N)}}. \tag{2}$$

Comme on a un cône

$$(A,\ \overline{\psi} : F \circ \overline{\delta} \to I_{\mathcal{D}}(A)),$$

il existe une unique flèche $\phi : G(\overline{\delta}) \to A$ de $\mathcal{D}$ telle que

$$I_{\mathcal{D}}(\phi) \circ \overline{\eta}_{F\circ\overline{\delta}} = \overline{\psi}. \tag{3}$$

Il reste à montrer que

$$I_{\mathcal{D}}(\phi) \circ \overline{\lambda} = \overline{\mu}$$
$$\Leftrightarrow \quad \forall N \in \text{Nœud}(\Delta^{\Phi}),\ \phi \circ \lambda_N = \mu_N : G(\overline{\Delta(N)}) \to A$$

Comme

$$(G(\overline{\Delta(N)}),\ \overline{\eta}_{F \circ \overline{\Delta(N)}} : F \circ \overline{\Delta(N)} \to I_{\mathcal{D}}(G(\overline{\Delta(N)})))$$

est cône colimite de $\overline{\Delta(N)}$, il suffit de montrer que

$$I_{\mathcal{D}}(\phi \circ \lambda_N) \circ \overline{\eta}_{F \circ \overline{\Delta(N)}} = I_{\mathcal{D}}(\mu_N) \circ \overline{\eta}_{F \circ \overline{\Delta(N)}}.$$

$$\begin{array}{lll} I_{\mathcal{D}}(\phi \circ \lambda_N) \circ \overline{\eta}_{F \circ \overline{\Delta(N)}} & & \\ \quad = \quad I_{\mathcal{D}}(\phi) \circ I_{\mathcal{D}}(\lambda_N) \circ \overline{\eta}_{F \circ \overline{\Delta(N)}} & & (I_{\mathcal{D}} \text{ foncteur}) \\ \quad = \quad I_{\mathcal{D}}(\phi) \circ \overline{\eta}_{F \circ \overline{\delta}} \circ \text{DIAGR}(F)(\overline{J_N}) & & (1) \\ \quad = \quad \overline{\psi} \circ \text{DIAGR}(F)(\overline{J_N}) & & (3) \\ \quad = \quad I_{\mathcal{D}}(\mu_N) \circ \overline{\eta}_{F \circ \overline{\Delta(N)}} & & (2) \end{array}$$

Unicité. Soit $\phi' : G(\overline{\delta}) \to A$ une flèche de $\mathcal{D}$ telle que

$$I_{\mathcal{D}}(\phi') \circ \overline{\lambda} = \overline{\mu}.$$

On montre que $\phi = \phi'$. Pour cela, il suffit de montrer que

$$I_{\mathcal{D}}(\phi') \circ \overline{\eta}_{F \circ \overline{\delta}} = \overline{\psi}$$
$$\Leftrightarrow \quad \forall (N, n) \in \text{Nœud}(\delta^{\Phi}),\ \phi'_{(N,n)} \circ (\eta_{F \circ \overline{\delta}})_{(N,n)} = \psi_{N,n}$$

$$\begin{array}{lll} \phi'_{(N,n)} \circ (\eta_{F \circ \overline{\delta}})_{(N,n)} & & \\ \quad = \quad \phi'_{(N,n)} \circ \lambda_N \circ (\eta_{F \circ \overline{\Delta(N)}})_n \circ F((J_N)_n^{-1}) & & (1) \\ \quad = \quad \mu_N \circ (\eta_{F \circ \overline{\Delta(N)}})_n \circ F((J_N)_n^{-1}) & & (\text{définition de } \phi') \\ \quad = \quad \psi_{N,n} & & (\text{définition de } \psi_{N,n}) \end{array}$$

□

Proposition 2.3 *Soit une catégorie $\mathcal{C}$ et une catégorie finiment cocomplète $\mathcal{D}$. Soit un foncteur $F : \mathcal{C} \to \mathcal{D}$. Posons*

$$G = \text{Colim}_{\mathcal{D}} \circ \text{DIAGR}(F) : \text{DIAGR}(\mathcal{C}) \to \mathcal{D}$$
$$H = \text{colim}_{\mathcal{D}} \circ \text{diagr}(F) : \text{diagr}(\mathcal{C}) \to \mathcal{D}.$$

Alors,

1. $H \circ [-]_{\mathcal{C}} = G$;
2. $H \circ [-]_{\mathcal{C}} \circ I_{\mathcal{C}} \cong F$;
3. *H conserve les colimites finies.*

Preuve. On prouve successivement les points 1, 2 et 3.

1. On a

$$\begin{array}{rcll} H \circ [-]_{\mathcal{C}} & = & \mathrm{colim}_{\mathcal{D}} \circ \mathrm{diagr}(F) \circ [-]_{\mathcal{C}} & (\text{définition de } H) \\ & = & \mathrm{colim}_{\mathcal{D}} \circ [-]_{\mathcal{D}} \circ \mathrm{DIAGR}(F) & (\text{définition de } \mathrm{diagr}(F)) \\ & = & \mathrm{Colim}_{\mathcal{D}} \circ \mathrm{DIAGR}(F) & (\text{définition de } \mathrm{colim}_{\mathcal{D}}) \\ & = & G & (\text{définition de } G) \end{array}$$

2. D'après le lemme 2.22, $G \circ I_{\mathcal{C}} \cong F$, donc, d'après le point 1, $H \circ [-]_{\mathcal{C}} \circ I_{\mathcal{C}} \cong F$.

3. Soit un diagramme $\overline{\overline{\Gamma}}$ de $\mathrm{DIAGR}(\mathrm{diagr}(\mathcal{C}))$. On considère un diagramme $\overline{\overline{\Delta}}$ de $\mathrm{DIAGR}^2(\mathcal{C})$ tel que

$$\overline{\overline{\Gamma}} = [-]_{\mathcal{C}} \circ \overline{\overline{\Delta}}.$$

On pose $\overline{\delta} = \mathrm{Apl}_{\mathcal{C}}(\overline{\overline{\Delta}})$, et le cône colimite du diagramme $\overline{\overline{\Gamma}}$

$$(\overline{\delta},\ \overline{\overline{\eta}} : \overline{\overline{\Gamma}} \to I_{\mathrm{diagr}(\mathcal{C})}(\overline{\delta}))$$

avec $\overline{\eta_N} = [\overline{J_N}]$.

On a

$$H \circ \overline{\overline{\Gamma}} = H \circ [-]_{\mathcal{C}} \circ \overline{\overline{\Delta}} = G \circ \overline{\overline{\Delta}}$$

et

$$\lambda_N = G(\overline{J_N}) = H([\overline{J_N}]) = H(\overline{\eta_N}).$$

D'après le lemme 2.24,

$$(G(\overline{\delta}),\ \overline{\lambda} : G \circ \overline{\overline{\Delta}} \to I_{\mathcal{D}}(G(\overline{\delta}))$$

est un cône colimite de $G \circ \overline{\overline{\Delta}}$, donc

$$(H(\overline{\delta}),\ H\overline{\overline{\eta}} : H \circ \overline{\overline{\Gamma}} \to I_{\mathcal{D}}(H(\overline{\delta})))$$

est un cône colimite du diagramme $H \circ \overline{\overline{\Gamma}}$. Par conséquent, H conserve les colimites finies.

□

Théorème 2.7 *(Complétion) Soit $\mathcal{C}$ une catégorie quelconque. La catégorie* $\mathrm{diagr}(\mathcal{C})$ *est une complétion de $\mathcal{C}$ par colimites finies. Autrement dit, pour toute catégorie finiment cocomplète $\mathcal{D}$, et tout foncteur $F : \mathcal{C} \to \mathcal{D}$, il existe un foncteur $H : \mathrm{diagr}(\mathcal{C}) \to \mathcal{D}$, unique à un isomorphisme naturel près, qui conserve les colimites finies, et tel que :*

$$H \circ [-]_{\mathcal{C}} \circ I_{\mathcal{C}} \cong F.$$

Preuve. Soit une catégorie $\mathcal{C}$ quelconque, et une catégorie $\mathcal{D}$ finiment cocomplète. Soit un foncteur $F : \mathcal{C} \to \mathcal{D}$. On montre l'existence, puis l'unicité à un isomorphisme près, d'un foncteur

$$H : \text{diagr}(\mathcal{C}) \to \mathcal{D}$$

qui conserve les colimites finies et tel que

$$H \circ [-]_{\mathcal{C}} \circ I_{\mathcal{C}} \cong F.$$

Existence. On pose $H = \text{colim}_{\mathcal{D}} \circ \text{diagr}(F)$. D'après la proposition 2.3, $H \circ [-]_{\mathcal{C}} \circ I_{\mathcal{C}} \cong F$ et H conserve les colimites finies.

Unicité de H à un isomorphisme naturel près.
Soit deux foncteurs $H, H' : \text{diagr}(\mathcal{C}) \to \mathcal{D}$ qui conservent les colimites finies, et tels que

$$H \circ [-]_{\mathcal{C}} \circ I_{\mathcal{C}} \cong H' \circ [-]_{\mathcal{C}} \circ I_{\mathcal{C}}.$$

On montre qu'il existe un isomorphisme naturel

$$H \cong H'.$$

$$\begin{array}{lll}
H \circ [-]_{\mathcal{C}} \circ I_{\mathcal{C}} \cong H' \circ [-]_{\mathcal{C}} \circ I_{\mathcal{C}} & & \\
\quad \Rightarrow \text{DIAGR}(H \circ [-]_{\mathcal{C}} \circ I_{\mathcal{C}}) \cong \text{DIAGR}(H' \circ [-]_{\mathcal{C}} \circ I_{\mathcal{C}}) & & \text{(lemme 2.6)} \\
\quad \Rightarrow \text{DIAGR}(H) \circ \text{DIAGR}([-]_{\mathcal{C}} \circ I_{\mathcal{C}}) \cong & & \\
\qquad\qquad \text{DIAGR}(H') \circ \text{DIAGR}([-]_{\mathcal{C}} \circ I_{\mathcal{C}}) & & \text{(DIAGR foncteur)} \\
\quad \Rightarrow \text{Colim}_{\mathcal{D}} \circ \text{DIAGR}(H) \circ \text{DIAGR}([-]_{\mathcal{C}} \circ I_{\mathcal{C}}) \cong & & \\
\qquad\qquad \text{Colim}_{\mathcal{D}} \circ \text{DIAGR}(H') \circ \text{DIAGR}([-]_{\mathcal{C}} \circ I_{\mathcal{C}}) & & \text{(propriété 2.3)} \\
\quad \Rightarrow H \circ \text{Colim}_{\text{diagr}(\mathcal{C})} \circ \text{DIAGR}([-]_{\mathcal{C}} \circ I_{\mathcal{C}}) \cong & & \\
\qquad\qquad H' \circ \text{Colim}_{\text{diagr}(\mathcal{C})} \circ \text{DIAGR}([-]_{\mathcal{C}} \circ I_{\mathcal{C}}) & & \text{(propriété 2.6)} \\
\quad \Rightarrow H \circ [-]_{\mathcal{C}} \cong H' \circ [-]_{\mathcal{C}} & & \text{(lemme 2.21)} \\
\quad \Rightarrow H \cong H' & & \text{(lemme 2.14)}
\end{array}$$

□

Remarque 2.11 Nous avons montré que la catégorie $\text{diagr}(\mathcal{C})$ est finiment cocomplète. Par contre, nous n'avons pas montré que $\text{diagr}(\mathcal{C})$ *a des colimites choisies.* En effet, nous n'avons pas montré l'existence d'une colimite *canonique* pour un objet de $\text{DIAGR}(\text{diagr}(\mathcal{C}))$, puisque pour montrer l'existence d'une colimite, nous devons choisir un représentant pour une classe d'équivalence de flèches de $\text{DIAGR}(\mathcal{C})$.

2.4 Conclusion

Dans ce chapitre, nous avons défini une catégorie de diagrammes $\text{DIAGR}(\mathcal{C})$, dont les objets sont les diagrammes finis, et les flèches sont les morphismes de diagrammes (définitions 2.16 et 2.18). Nous avons rappelé la définition de *colimite* d'un diagramme et de colimite d'un morphisme de diagrammes (paragraphe 2.3.2). Nous avons défini une opération d'*aplatissement* $\text{Apl}_{\mathcal{C}} : \text{DIAGR}^2(\mathcal{C}) \to \text{DIAGR}(\mathcal{C})$ qui

permet de considérer tout diagramme de diagrammes comme un simple diagramme (définition 2.29).

Nous avons défini une relation de congruence sur les morphismes de diagrammes, ce qui nous a permis de considérer la catégorie quotient diagr($\mathcal{C}$) (définition 2.32). Nous avons montré que la catégorie diagr($\mathcal{C}$) est finiment cocomplète, c'est-à-dire que tout diagramme fini sur diagr($\mathcal{C}$) a une colimite dans diagr($\mathcal{C}$) (théorème 2.6). Cette colimite d'un objet $\overline{\overline{\Gamma}}$ de DIAGR(diagr($\mathcal{C}$)) peut être construite en *aplatissant* un diagramme $\overline{\overline{\Delta}}$ de DIAGR^2($\mathcal{C}$) obtenu à partir de $\overline{\overline{\Gamma}}$ en remplaçant chaque classe d'équivalence qui étiquète un arc par un représentant de cette classe.

Enfin, nous avons montré que la catégorie diagr($\mathcal{C}$) est une complétion de $\mathcal{C}$ par colimites finies (théorème 2.7).

Références bibliographiques

[AL91] A. Asperti and G. Longo. *Categories, Types and Structures, An Introduction to Category Theory for the Working Computer Scientist.* Foundations of Computing Science, MIT Press, 1991.

[AM75] M. A. Arbib and E. G. Manes. *Arrows, Structures and Functors: The Categorical Imperative.* Academic Press, 1975.

[B+90] D. Bert et al. Reference manual of the specification language LPG. Technical Report 59, LIFIA, Mars 1990. Anonymous ftp at `imag.fr`, in `/pub/SCOP/LPG/NewSun4/man_lpg.dvi`.

[BBC86] G. Bernot, M. Bidoit, and C. Choppy. Abstract data types with exception handling: An initial approach based on a distinction between exception and errors. *Theoretical Computer Science*, 46(1):13–45, 1986.

[BDMN73] G. Birtwistle, O.-J. Dahl, B. Myrhaug, and K. Nygaard. *Simula Begin.* Auerbach Pub., New York, 1973.

[BE86] D. Bert and R. Echahed. Design and implementation of a generic, logic and functional programming language. In *Proceedings of ESOP'86*, number 213 in LNCS, pages 119–132. Springer-Verlag, 1986.

[Ber83] D. Bert. Refinements of generic specifications with algebraic tools. In *Proceedings of IFIP'83, Paris*, pages 815–820, 1983.

[Ber90] D. Bert. Spécification de logiciels réutilisables. Technical Report RR-828-I-IMAG-116, LIFIA, Octobre 1990.

[BG77] R. M. Burstall and J. A. Goguen. Putting theories together to make specifications. In *Proceedings of the 5^{th} International Joint Conference on Artificial Intelligence*, pages 1045–1058, 1977.

[BG80] R. M. Burstall and J. A. Goguen. The semantics of CLEAR, a specification language. In *Proceedings of Advanced Course on Abstract Software Specification*, number 86 in LNCS, pages 292–332. Springer-Verlag, 1980.

[BHK90] J. A. Bergstra, J. Heering, and R. Klint. Module algebra. *J. ACM*, 37(2):335–372, Apr. 1990.

[Bid89] M. Bidoit. PLUSS, un langage pour le développement de spécifications algébriques modulaires, Mai 1989. Thèse d'État, Université de Paris-Sud.

[BL93] G. Bernot and P. Le Gall. Exception handling and term labelling. In *Proceedings of TAPSOFT'93*, number 668 in LNCS, pages 421–436. Springer-Verlag, 1993.

[BO95] D. Bert and C. Oriat. A model inference system for generic specification with application to code sharing. In *Proceedings of TAPSOFT'95*, number 915 in LNCS, pages 741–755. Springer-Verlag, 1995.

[BR86] R. M. Burstall and D. Rydeheard. Computing with categories. Technical Report ECS-LFCS-86-9, University of Edinburgh, September 1986.

[Bur80] R. M. Burstall. Electronic category theory. In *Proceedings of the 9^{th} Symposium on Mathematical Foundations of Computer Science*, 1980.

[BW85] S. L. Bloom and E. G. Wagner. Many-sorted theories and their algebras with some applications to data types. In M. Nivat and J. C. Reynolds, editors. *Algebraic Methods in Semantics*, chapter 4, pages 133–168. Cambridge University Press, 1985.

[BW90] M. Barr and C. Wells. *Category Theory for Computing Science.* Prentice-Hall International, 1990.

[BW94] M. Barr and C. Wells. The categorical theory generated by a limit sketch, Nov. 1994.

[Car86] J. Cartmell. Generalized algebraic theories and contextual categories. *Annals of Pure and Applied Logic*, 32:209–243, 1986.

[Coh65] P. M. Cohn. *Universal Algebra.* Harper and Row, 1965. Revised version 1980.

[Cur91] F. Cury. Catégories lax-localement-cartésiennes et catégories localement cartésiennes: un exemple de suffisante complétude connexe de sémantiques initiales. In *diagrammes*, volume 25, pages 1–155, Université Paris 7, Juillet 1991.

[DN66] O.-J. Dahl and K. Nygaard. Simula – an Algol-based simulation language. *Communications of the ACM*, 9:671–678, Sept. 1966.

[DR94a] D. Duval and J.-C. Reynaud. Sketches and computation (part 1): Basic definitions and static evaluation. *Mathematical Structures in Computer Science*, 4:185–238, 1994.

[DR94b] D. Duval and J.-C. Reynaud. Sketches and computation (part 2): Dynamic evaluation and applications. *Mathematical Structures in Computer Science*, 4:239–271, 1994.

[Ehr68] C. Ehresmann. Esquisses et types de structures algébriques. *Bulletin de l'Institut Polytechnique*, Iasi 14, 1968.

[EJO93] H. Ehrig, R. M. Jimenez, and F. Orejas. Compositionality results for different types of parameterization and parameter passing in specification languages. In *Proceedings of the 4^{th} International Joint Conference CAAP/FASE*, number 668 in LNCS, pages 31–45. Springer-Verlag, 1993.

[EM85] H. Ehrig and B. Mahr. *Fundamentals of Algebraic Specification 1. Equations and Initial Semantics*, volume 6 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, 1985.

[EM90] H. Ehrig and B. Mahr. *Fundamentals of Algebraic Specification 2. Module Specifications and Constraints*, volume 21 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, 1990.

[FGJM85] K. Futatsugi, J. A. Goguen, J.-P. Jouannaud, and J. Meseguer. Principles of OBJ2. In *Proceedings of Principles of Programming Languages*, pages 52–66, 1985.

[Gau84] M.-C. Gaudel. A first introduction to PLUSS. Technical report, Université d'Orsay, France, 1984.

[GB84] J. A. Goguen and R. M. Burstall. Introducing institutions. In *Proceedings of the Workshop on Logic of Programming*, number 164 in LNCS, pages 221–256. Springer-Verlag, 1984.

[GB90] J. A. Goguen and R. M. Burstall. Institutions: Abstract model theory for specification and programming. Research Report ECS-LFCS-90-106, University of Edinburgh, January 1990.

[GKK+87] J. A. Goguen, C. Kirchner, H. Kirchner, A. Mégrelis, J. Meseguer, and T. Winkler. An introduction to OBJ3. In *Proceedings of the 1^{st} International Workshop on Conditional Term Rewriting Systems*, number 308 in LNCS, pages 258–263. Springer-Verlag, 1987.

[Gog73] J. A. Goguen. Categorical foundations for general systems theory. In *Advances in Cybernetics and System Research*, pages 121–130. Transcripta Books, 1973.

[Gog92] J. A. Goguen. Sheaf semantics for concurrent interacting objects. *Mathematical Structures in Computer Science*, 2:159–191, 1992.

[Gol79] R. Goldblatt. *Topoi: The Categorial Analysis of Logic*, volume 98 of *Studies in Logic and the Foundations of Mathematics*. North Holland, 1979.

[GTW78] J. A. Goguen, J. W. Thatcher, and E. G. Wagner. An initial algebra approach to the specification, correctness, and implementation of abstract data types. In R. T. Yeh, editor, *Current Trends in Programming*

Methodology, volume 4: Data Structuring, pages 80–149. Prentice-Hall, 1978.

[GTWW75] J. A. Goguen, J. W. Thatcher, E. G. Wagner, and J. B. Wright. Abstract data types as initial algebras and the correctness of data representation. In *Computer Graphics, Pattern Recognition and Data Structure*, pages 89–93, 1975.

[GTWW77] J. A. Goguen, J. W. Thatcher, E. G. Wagner, and J. B. Wright. Initial algebra semantics and continuous algebra. *J. ACM*, 24:68–95, 1977.

[Gut75] J. V. Guttag. *The specification and application to programming of abstract data types.* PhD thesis, University of Toronto, 1975.

[Gut77] J. V. Guttag. Abstract data types and the development of data structures. *Communication of the ACM*, 6:396–404, 1977.

[Huf92] J.-M. Hufflen. Proposal for GLIDER version 1.0: Principles and main features. ICARUS Technical Report, INRIA-LORRAINE & CRIN, 1992.

[Law63] W. Lawvere. Functorial semantics of algebraic theories. *Proc. Nat. Acad. Sci.*, 50:869–873, 1963.

[LCW85] D. Lorge Parnas, P. C. Clements, and D. M. Weiss. The modular structure of complex systems. *IEEE Transactions on Software Engineering*, SE-11(3):259–266, March 1985.

[LS86] J. Lambek and P. J. Scott. *Introduction to higher order categorical logic.* Cambridge studies in advanced mathematics, 1986.

[LZ74] B. Liskov and S. Zilles. Programming with abstract data types. *ACM SIGPLAN Notices*, 9(4):50–59, 1974.

[Mar95] A. Martins. *La généralisation : un outil pour la réutilisation.* PhD thesis, INPG, Grenoble, Mars 1995.

[MB70] S. Mac Lane and G. Birkhoff. *Algèbre. 1. Structures fondamentales.* Gauthier-Villars, 1970.

[McL71] S. Mac Lane. *Categories for the Working Mathematician.* Springer-Verlag, 1971.

[MG85] J. Meseguer and J. A. Goguen. Initiality, induction, and computability. In M. Nivat and J. C. Reynolds, editors, *Algebraic Methods in Semantics*, chapter 14, pages 459–541. Cambridge University Press, 1985.

[Ori94] C. Oriat. Representing modular specifications as diagrams. In *Compte Rendu des Journées du GDR Programmation, Lille*, pages 53–68, Septembre 1994.

[Ori95] C. Oriat. Detecting isomorphisms of modular specifications with diagrams. In *Proceedings of AMAST'95*, number 936 in LNCS, pages 184–198. Springer-Verlag, 1995.

[Poi92] A. Poigné. Basic category theory. In *Handbook of Logic in Computer Science, Volume 1. Background: Mathematical Structures*, pages 413–640. Oxford Science Publication, 1992.

[Ren91] G. Renardel de Lavalette. Logical semantics of modularisation. In *Proceedings of CSL'91*, number 626 in LNCS, pages 306–315. Springer-Verlag, 1991.

[Rey87] J.-C. Reynaud. Sémantique de LPG. Research Report 651 I IMAG, LIFIA, Mars 1987.

[Rey90a] J.-C. Reynaud. Putting algebraic components together: A dependent type approach. Research Report 810 I IMAG, LIFIA, Avril 1990. Extended version.

[Rey90b] J.-C. Reynaud. Putting algebraic components together: A dependent type approach. In *Proceedings of DISCO'90*, number 429 in LNCS. Springer-Verlag, 1990.

[Rey93] J.-C. Reynaud. Isomorphism of typed algebraic specifications. Internal Report, LGI-IMAG, Avril 1993.

[See84] R. A. G. Seely. Locally cartesian closed categories and type theory. *Math. Proc. Camb. Phil. Soc.*, 95(33):33–48, 1984.

[SP77] M. B. Smyth and G. D. Plotkin. The category-theoretic solution of recursive domain equations. In *Proceedings of the 18^{th} IEEE Symposium on Foundations of Computer Science*, pages 13–17, 1977.

[SST92] D. Sannella, S. Sokolowski, and A. Tarlecki. Toward formal development of programs from algebraic specifications: Parameterisation revisited. *Acta Informatica*, 29:689–736, 1992.

[ST88] D. Sannella and A. Tarlecki. Specifications in an arbitrary institution. *Information and Computation*, 76:165–210, 1988.

[SW83] D. Sannella and M. Wirsing. A kernal language for algebraic specification and implementation. In *Proceedings of the 11^{th} Colloquium on Foundations of Computation Theory*, number 158 in LNCS, pages 413–427. Springer-Verlag, 1983.

[SW91] T. Streicher and M. Wirsing. Dependent types considered necessary for specification languages. In *Proceedings of the 7^{th} Workshop on Specification of Abstract Data Types*, number 534 in LNCS, pages 323–339, 1991.

[TBG91] A. Tarlecki, R. M. Burstall, and J. A. Goguen. Some fundamental algebraic tools for the semantics of computation: Part 3. Indexed categories. *Theoretical Computer Science*, 91:239–264, 1991.

[TWW82] J. W. Thatcher, E. G. Wagner, and J. B. Wright. Data type specification: Parameterization and the power of specification techniques. *ACM Trans. Prog. Lang. Syst.*, 4:711–773, 1982.

[Wan79] M. Wand. Fixed-point constructions in order-enriched categories. *Theoretical Computer Science*, 8:13–30, 1979.

[WBT85] E. G. Wagner, S. L. Bloom, and J. W. Thatcher. Why algebraic theories? In M. Nivat and J. C. Reynolds, editors, *Algebraic Methods in Semantics*, chapter 17, pages 607–634. Cambridge University Press, 1985.

[Wir86] M. Wirsing. Structured algebraic specifications: A kernel language. *Theoretical Computer Science*, 42:123–249, 1986.

[Wir90] M. Wirsing. Algebraic specification. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, chapter 13, pages 677–788. Elsevier Science Publishers B.V., 1990.

[Wir94] M. Wirsing. Algebraic specification languages: An overview. In *Recent Trends in Data Type Specification. 10^{th} Workshop on Specification of Abstract Data Types*, number 906 in LNCS. 1994.

Table de la première partie

Introduction **9**
1 Spécifications algébriques 9
2 Théorie des catégories 10
3 Modularité 11
4 Notre travail 12
5 Comparaisons avec d'autres travaux 14
6 Plan de ce mémoire 16

1 Motivation **19**
1.1 Préliminaires 19
1.2 Spécifications 25
1.3 Composition de spécifications en LPG 29
1.4 Sommes amalgamées 31
1.4.1 Somme amalgamée de deux ensembles 32
1.4.2 Somme amalgamée de deux signatures 34
1.4.3 Somme amalgamée de deux spécifications 36
1.4.4 Syntaxe abstraite pour les constructions modulaires 38
1.4.5 Somme disjointe de deux spécifications 39
1.5 Spécifications modulaires d'anneaux 40
1.6 Isomorphisme de construction 46
1.7 Cadre général 47
1.7.1 Institutions 47
1.7.2 Terminologie et hypothèses 49

2 Graphes, catégories et diagrammes **51**
2.1 Graphes 51
2.2 Catégories, foncteurs 53
2.2.1 Catégories 54
2.2.2 Foncteurs 55
2.2.3 Transformations naturelles 56
2.2.4 Adjonctions 59
2.3 Diagrammes 61
2.3.1 Catégorie des diagrammes DIAGR($\mathcal{C}$) 61
2.3.2 Colimites 68
2.3.3 Exemples de colimites 72

2.3.4 Adjonction ($\mathrm{Colim}_{\mathcal{C}} \dashv I_{\mathcal{C}}$) 75
2.3.5 Conservation de colimites 77
2.3.6 Aplatissement 83
2.3.7 Catégorie diagr($\mathcal{C}$) 89
2.3.8 La catégorie diagr($\mathcal{C}$) est finiment cocomplète 99
2.3.9 La catégorie diagr($\mathcal{C}$) est une complétion de $\mathcal{C}$ par colimites finies 105
2.4 Conclusion 110

Références bibliographiques **113**

Laboratoire LSR (Logiciels, Systèmes, Réseaux)
Institut IMAG
(Institut d'Informatique et de Mathématiques Appliquées de Grenoble)
681, rue de la Passerelle
B.P. 72
38402 Saint Martin d'Hères Cedex