SYSTÈMES D'INFÉRENCE NON MONOTONE

Philippe BESNARD*

1. Introduction

Concernant la formalisation du raisonnement, le premier sujet d'études a été la logique classique, dont voici une brève description de rappel. Le langage de la logique classique est basé sur un vocabulaire comportant une infinité de variables et une liste de symboles de relations (prédicats), avec des connecteurs $\neg$ (négation), $\rightarrow$ (implication), $\wedge$ (conjonction), $\vee$ (disjonction) ainsi que les quantificateurs $\forall$ (quel que soit) et $\exists$ (il existe). Les formules sont obtenues d'abord en appliquant des variables aux prédicats puis par composition à l'aide des connecteurs et quantificateurs. Il est possible d'avoir d'autres termes que les variables en arguments des prédicats, en ayant recours à des constantes et des symboles de fonctions. Les formules ont, généralement, une lecture assez intuitive, par exemple $\neg\exists x\ P(x,x)$ signifie que P est une relation antiréflexive et $\forall xy\ P(x,y) \rightarrow P(y,x)$ indique que P est symétrique.

Pour ce qui est de la sémantique de la logique classique, la notion de base est l'interprétation, une structure basée sur un ensemble non vide appelé domaine. Dans une interprétation, une fonction (respectivement relation) sur le domaine de l'interprétation est associée à chaque symbole de fonction (respectivement relation). La vérité d'une formule dans une interprétation est évaluée d'après la valeur associée aux symboles de fonction et de relation apparaissant dans la formule. Par exemple, $\forall xyz\ P(x,y) \wedge P(y,z) \rightarrow P(x,z)$ est évaluée à vrai dans toute interprétation qui associe à P une relation transitive. La notion d'inférence qui se dégage de la sémantique de la logique classique se définit ainsi: s'ensuit des prémisses toute formule vraie dans chaque interprétation où toutes les prémisses sont vraies (une telle interprétation étant appelée un modèle des prémisses). Ainsi, de $\neg\exists xy\ P(x,y)$ (le fait que P soit vide) s'ensuit $\neg\forall x\ P(x,x)$ (le fait que P n'est pas réflexive), ce qui sera noté $\neg\exists xy\ P(x,y) \models \neg\forall x\ P(x,x)$.

Il existe différents systèmes de règles, tous équivalents à l'approche sémantique, qui permettent de formaliser des déductions en logique classique (c'est-à-dire de calculer si une certaine formule s'ensuit d'un ensemble donné de formules). Par exemple, un tel système de règles permet d'établir que $\forall x\ P(x,x)$ (le fait que P soit réflexive) s'ensuit de $\forall xy\ P(x,y) \vee P(y,x)$ (le fait que P soit totale), ce qui sera noté $\forall xy\ P(x,y) \vee P(y,x) \vdash \forall x\ P(x,x)$ (la notation $Th(\mathcal{A}) = \{F \mid \mathcal{A} \vdash F\}$ sera également utilisée dans la suite).

Du point de vue de la formalisation du raisonnement, le fait que la logique classique soit un système formel a permis de déterminer certaines limitations à ses possibilités. Notamment, que la logique classique ne permet pas de représenter

THEORIA - Segunda Época - Vol. X
1995, Nº. 23, 37-47

le concept de vérité (Tarski 1939) ni de nécessité (Montague 1963). Et encore, que les raisonnements de type conjecturel ne pouvaient être formalisés par la logique classique. Par exemple, si deux personnes prennent chacune un nombre au hasard, il est raisonnable de conjecturer qu'il ne s'agit pas du même. Ce genre de raisonnement s'avère aussi fécond que fréquent en pratique. D'où l'intérêt de développer l'étude logique de ce type de raisonnements, le point de départ étant l'étude de la monotonie, la propriété de la logique classique qui exclut toute possibilité de modélisation en logique classique de tels raisonnements.

2. Caractérisation de classes de systèmes non-monotones

2.1. Approche axiomatique

Les propiétés «structurelles» de la logique classique ont été mises à jour par Tarski (1930) (qui les assimile aux propriétés caractéristiques des opérateurs de fermeture) et Gentzen (1933). Elles peuvent se résumer en deux clauses ($\mathfrak{T}$, $\mathfrak{S}$ et $\mathfrak{U}$ sont des ensembles de formules, A une formule, la virgule dénote l'adjonction de formules à un ensemble de formules):

Réflexivité: si $A \in \mathfrak{T}$ alors $\mathfrak{T} \vdash A$

Coupure: si $\mathfrak{T} \vdash \mathfrak{U}$ et $\mathfrak{S}, \mathfrak{U} \vdash A$ alors $\mathfrak{T}, \mathfrak{S} \vdash A$

Il convient sans doute de mentionner dès à présent que dans la formulation de toute propriété, il revient au même de mettre une simple formule F à la place de $\mathfrak{U}$ à condition que le système considéré vérifie la:

Compacité descendante: si $\mathfrak{T} \vdash A$ alors il existe une partie finie $\mathfrak{S}$ de $\mathfrak{T}$ telle que $\mathfrak{S} \vdash A$

En réalité, la réflexivité et la coupure ne sont pas propres à la logique classique car elles définissent la classe des relations de conséquence, dont la logique classique fait partie. Un cas dégénéré (où $\mathfrak{U}$ est vide mais qui peut aussi s'obtenir à partir de la réflexivité et de la coupure avec $\mathfrak{U}$ valant A) de la coupure est la:

Monotonie: si $\mathfrak{S} \vdash A$ alors $\mathfrak{T}, \mathfrak{S} \vdash A$

La monotonie est incompatible avec l'aspect évolutif du raisonnement conjecturel: une conjecture peut être abandonnée alors que la monotonie exprime qu'une conclusion ne peut être remise en cause sous aucune circonstance. Il s'agit donc de définir des logiques conservant des propriétés structurelles significatives indépendantes de la monotonie (diverses logiques de ce type ayant été proposées (McCarthy 1980) (Reiter 1980) (Moore 1983) avant que l'étude de telles propriétés ne soit entreprise par (Gabbay 1985) (Besnard 1988) (Makinson 1989) (Kraus, Lehmann & Magidor 1990)). Un moindre affaiblissement des propriétés structurelles s'obtient en exigeant la:

Cumulativité: si $\mathfrak{T} \mid\!\sim \mathfrak{U}$ alors $\mathfrak{T} \mid\!\sim A$ ssi $\mathfrak{T}, \mathfrak{U} \mid\!\sim A$

Le symbole $\mid\!\sim$ est introduit pour signaler que la monotonie peut n'être pas satisfaite, le symbole $\vdash$ étant dorénavant réservé aux seules relations de conséquence (lesquelles sont, comme indiqué plus haut, caractérisées par la réflexivité et la coupure). La cumulativité, qui est due à Makinson, regroupe la

Monotonie affaiblie: si $\mathfrak{T} \mid\!\sim \mathfrak{U}$ et $\mathfrak{T} \mid\!\sim A$ alors $\mathfrak{T},\mathfrak{U} \mid\!\sim A$
(proposée à l'origine par Gabbay) et la

Transitivité: si $\mathfrak{T} \mid\!\sim \mathfrak{U}$ et $\mathfrak{T},\mathfrak{U} \mid\!\sim A$ alors $\mathfrak{T} \mid\!\sim A$
qui n'est rien d'autre qu'une instance de la coupure.

La cumulativité exprime que la logique considérée est stable pour la notion d'inférence développée: tout ensemble de conclusions $\mathfrak{U}$ peut être intégré aux prémisses $\mathfrak{T}$ sans qu'aucune conclusion A de $\mathfrak{T}$ ne soit perdue (monotonie affaiblie) ou gagnée (transitivité). Moyennant la compacité, la cumulativité peut d'ailleurs s'écrire sous la forme finitaire suivante avec n non nul quelconque:

si $\mathfrak{T} \mid\!\sim A_1$ et $\mathfrak{T} \mid\!\sim A_2$ et ... et $\mathfrak{T} \mid\!\sim A_n$ alors $\mathfrak{T},A_1,\ldots,A_n \mid\!\sim B$ ssi $\mathfrak{T} \mid\!\sim B$

La compacité étant encore posée, et en présence de la réflexivité, une autre formulation finitaire de la cumulativité est aussi évidente:

Aggrégation: si $\mathfrak{T} \mid\!\sim A_1$ et $\mathfrak{T},A_1 \mid\!\sim A_2$ et $\mathfrak{T},A_1,A_2 \mid\!\sim A_3$ et ... et $\mathfrak{T},A_1,\ldots,A_{n-1} \mid\!\sim A_n$ alors $\mathfrak{T},A_1,\ldots,A_n \mid\!\sim B$ ssi $\mathfrak{T} \mid\!\sim B$

Beaucoup moins immédiat est le fait que la cumulativité correspond à l'équivalence de l'intégration individuelle de prémisses qui s'obtiennent circulairement les unes des autres (Kraus, Lehmann & Magidor 1990):

Circularité: si $\mathfrak{T},\mathfrak{S}_0 \mid\!\sim \mathfrak{S}_1$ et $\mathfrak{T},\mathfrak{S}_1 \mid\!\sim \mathfrak{S}_2$ et ... et $\mathfrak{T},\mathfrak{S}_{n-1} \mid\!\sim \mathfrak{S}_n$ et $\mathfrak{T},\mathfrak{S}_n \mid\!\sim \mathfrak{S}_0$ alors $\mathfrak{T},\mathfrak{S}_0 \mid\!\sim A$ ssi $\mathfrak{T},\mathfrak{S}_1 \mid\!\sim A$ ssi ... ssi $\mathfrak{T},\mathfrak{S}_n \mid\!\sim A$

Une variante superficielle est:

si $\mathfrak{T} \mid\!\sim \mathfrak{S}_0$ et $\mathfrak{T},\mathfrak{S}_0 \mid\!\sim \mathfrak{S}_1$ et $\mathfrak{T},\mathfrak{S}_1 \mid\!\sim \mathfrak{S}_2$ et ... et $\mathfrak{T},\mathfrak{S}_{n-1} \mid\!\sim \mathfrak{S}_n$ et $\mathfrak{T},\mathfrak{S}_n \mid\!\sim \mathfrak{S}_0$ alors $\mathfrak{T} \mid\!\sim A$ ssi $\mathfrak{T},\mathfrak{S}_1 \mid\!\sim A$ ssi ... ssi $\mathfrak{T},\mathfrak{S}_n \mid\!\sim A$

La forme finitaire (supposant la compacité et la réflexivité) la plus intéressante de cette formulation de la cumulativité correspond au cas n=2:

si $\mathfrak{T} \mid\!\sim A$ et $\mathfrak{T},A \mid\!\sim B$ et $\mathfrak{T},B \mid\!\sim A$ alors $\mathfrak{T} \mid\!\sim C$ ssi $\mathfrak{T},B \mid\!\sim C$

La cumulativité c'est enfin si $\mathfrak{T} \mid\!\sim \mathfrak{S}_0$ et ... et $\mathfrak{T} \mid\!\sim \mathfrak{S}_n$ alors $\mathfrak{T},\mathfrak{S}_0 \mid\!\sim A$ ssi ... ssi $\mathfrak{T},\mathfrak{S}_n \mid\!\sim A$

Les différentes formulations listées ci-dessus montrent que la cumulativité est assez riche du point de vue propriétés. L'objectif qui était de s'affranchir de la monotonie dans la théorie des systèmes logiques sans tomber sur un appauvrissement radical de la théorie peut donc être considéré comme atteint. Il reste malgré tout à vérifier que la théorie obtenue rend compte de l'essentiel des raisonnements qu'il s'agissait de modéliser au départ (et qui ont motivé l'étude de logiques censées relever de la théorie remaniée). Autrement dit, les logiques cumulatives permettent-elles de formaliser la majeure partie des raisonnements de type conjecturel? Même si l'expression «la majeure partie» est prise en un sens pas trop fort, la réponse est malheureusement négative.

Les raisonnements dans lesquels intervient une certaine notion d'ordre de grandeur échappent à la formalisation par une logique cumulative. Par exemple, étant admis que tout ce qui a une probabilité d'au moins 2/3 est vraisemblable et sachant qu'un élément d'un ensemble fini a 2/3 de chances de se trouver dans les deux premiers tiers d'une énumération quelconque de l'ensemble, il est normal de conclure qu'un élément **e** donné se trouve vraisemblablement dans les deux premiers tiers de l'énumération, ce qui est noté $\mathfrak{T} \mid\!\sim P$. Il est tout aussi normal de conclure que l'élément **e** se trouve vraisemblablement dans les deux derniers tiers de l'énumération, ce qui est noté $\mathfrak{T} \mid\!\sim D$. Mais au cas où il est vraiment établi que **e** se situe dans les deux premiers tiers de l'énumération, il n'est plus question de conclure que **e** se trouve vraisemblablement dans les deux derniers tiers. La cumulativité n'est pas respectée puisque $\mathfrak{T} \mid\!\sim P$ et $\mathfrak{T} \mid\!\sim D$ bien que $\mathfrak{T}, P \not\mid\!\sim D$. Il n'y a clairement rien de vicié dans ce raisonnement, en particulier parce que les propositions P et D ne sont pas contradictoires (en fait, la vraisemblance de deux événements n'implique pas la vraisemblance de leur conjonction).

Le cas qui vient d'être exposé constitue un contre-exemple à la monotonie affaiblie. Dans la même veine, un contre-exemple à la transitivité est le suivant. Il est normal de conclure *a priori* que l'élément **e** est vraisemblablement dans le deux premiers tiers de l'énumération et, au cas où il est connu que **e** s'y trouve effectivement, de conclure que **e** se situe vraisemblablement dans la première moitié de l'énumération, ce qui est noté $\mathfrak{T} \mid\!\sim P$ et $\mathfrak{T}, P \mid\!\sim M$. Mais il est hors de question de conclure *a priori* que **e** se situe vraisemblablement dans la première moitié de l'énumération: $\mathfrak{T} \not\mid\!\sim M$.

Plus généralement, il apparaît que c'est la technique de «seuil» qui est en cause dans la violation de la cumulativité par des raisonnements faisant intervenir des degrés numériques mais ce point ne sera pas développé davantage ici.

La cumulativité n'est finalement pas satisfaite par beaucoup des systèmes non-monotones existants (Besnard 1988). Pour effectuer une analyse plus fine de ces systèmes, il faut recourir à des propriétés moins fortes que la cumulativité. Quant à savoir comment dégrader la cumulativité, une solution se dessine à partir du constat selon lequel les systèmes non-monotones existants s'appuient sur une notion de déduction sous-jacente mise en œuvre par une relation de conséquence. C'est ainsi qu'une version atténuée de la cumulativité pourrait également faire appel à la relation de conséquence sous-jacente, comme illustré par ce qui suit.

Pour une relation non-monotone $\mid\!\sim$ développée à partir d'une relation $\vdash$ satisfaisant elle la réflexivité et la coupure, il est clair que la cumulativité

si $\mathfrak{T} \mid\!\sim \mathfrak{A}$ alors $\mathfrak{T} \mid\!\sim A$ ssi $\mathfrak{T}, \mathfrak{A} \mid\!\sim A$

peut être relaxée (Besnard 1988) en:

Cumulativité restreinte: si $\mathfrak{T} \vdash \mathfrak{A}$ alors $\mathfrak{T} \mid\!\sim A$ ssi $\mathfrak{T}, \mathfrak{A} \mid\!\sim A$

ou encore en:

si $\mathfrak{T} \mid\!\sim \mathfrak{A}$ et $\mathfrak{T} \vdash A$ alors $\mathfrak{T}, \mathfrak{A} \mid\!\sim A$

si $\mathfrak{T} \mid\!\sim \mathfrak{U}$ et $\mathfrak{T},\mathfrak{U} \vdash A$ alors $\mathfrak{T} \mid\!\sim A$

La cumulativité restreinte exprime que la relation $\mid\!\sim$ est stable relativement à la relation $\vdash$ au sens où deux ensembles de prémisses qui sont équivalents par $\vdash$ le sont aussi par $\mid\!\sim$ (c'est-à-dire que $\mid\!\sim$ leur associe le même ensemble de conclusions). En d'autres termes, la relation $\mid\!\sim$ n'est sensible aux variations syntaxiques que si elles sont significatives pour la relation $\vdash$ sous-jacente.

La complétion de prédicats (Clark 1978) est un système non-monotone qui ne respecte pas la cumulativité restreinte. Pour la complétion de predicats, $\vdash$ est la logique classique et $\mid\!\sim$ résulte du remplacement des prémisses sous la forme

$\forall x_1 \ldots x_n \; C_1(x_1 \ldots x_n) \vee \ldots \vee C_m(x_1 \ldots x_n) \rightarrow P(x_1 \ldots x_n)$
par $\forall x_1 \ldots x_n \; C_1(x_1 \ldots x_n) \vee \ldots \vee C_m(x_1 \ldots x_n) \leftrightarrow P(x_1 \ldots x_n)$.

A partir de là, un cas de violation de la cumulativité restreinte se construit facilement au vu de $\forall x(\exists y \neg P(y)) \rightarrow P(x) \vdash \forall x \; x{=}x \rightarrow P(x)$, par exemple. En effet, la complétion de prédicats de $\forall x(\exists y \neg P(y)) \rightarrow P(x)$ est contradictoire, c'est $\forall x(\exists y \neg P(y)) \leftrightarrow P(x)$, et la complétion de prédicats de $\{\forall x(\exists y \neg P(y)) \rightarrow P(x), \; \forall x \; x{=}x \rightarrow P(x)\}$ est $\forall x \; x{=}x \vee \exists y \neg P(y) \leftrightarrow P(x)$, qui n'est elle pas contradictoire. Cet exemple montre que la complétion de prédicats ne satisfait pas:

si $\mathfrak{T} \vdash \mathfrak{U}$ et $\mathfrak{T} \mid\!\sim A$ alors $\mathfrak{T},\mathfrak{U} \mid\!\sim A$

mais la complétion de prédicats respecte l'autre partie de la cumulativité restreinte, à savoir:

si $\mathfrak{T} \vdash \mathfrak{U}$ et $\mathfrak{T},\mathfrak{U} \mid\!\sim A$ alors $\mathfrak{T} \mid\!\sim A$.

Que la cumulativité restreinte soit falsifiée par la complétion de prédicats est rhédibitoire pour celle-ci: il apparaît pour le moins problématique d'attacher une signification satisfaisante à une notion d'inférence qui ne se plie pas à la cumulativité restreinte (une telle notion d'inférence est fermée sous une certaine relation de déduction sans respecter l'équivalence déductive associée à cette relation). En fait, la cumulativité restreinte est une propriété aussi importante que les deux autres proprietés énoncées à sa suite, c'est-à-dire:

(i) si $\mathfrak{T} \mid\!\sim \mathfrak{U}$ et $\mathfrak{T} \vdash A$ alors $\mathfrak{T},\mathfrak{U} \mid\!\sim A$
(ii) si $\mathfrak{T} \mid\!\sim \mathfrak{U}$ et $\mathfrak{T},\mathfrak{U} \vdash A$ alors $\mathfrak{T} \mid\!\sim A$

L'intérêt de la première d'entre elles, la propriété (i), devient absolument évident lorsqu'elle est mise sous la forme de la:

Conservation: si $\mathfrak{T} \vdash A$ alors $\mathfrak{T} \mid\!\sim A$

De manière assez triviale, l'équivalence avec la formulation (i) repose sur la monotonie de la relation $\vdash$. Tout aussi clairement, la conservation entraîne la réflexivité (puisque la relation $\vdash$ étant une relation de conséquence, satisfait elle-même la réflexivité). Plus substantiel est le fait que la conservation est conséquence de la propriété (ii):

Transitivité composée: si $\mathfrak{C} \mathrel{|\!\sim} \mathfrak{A}$ et $\mathfrak{C},\mathfrak{A} \vdash A$ alors $\mathfrak{C} \mathrel{|\!\sim} A$

La transitivité composée est du même niveau que la cumulativité restreinte au sens où la notion d'équivalence associée à la relation de conséquence $\vdash$ sous-jacente est intégrée à la relation $\mathrel{|\!\sim}$ par la cumulativité restreinte pour ce qui est des prémisses et par la transitivité composée pour ce qui est des conclusions.

En ce qui concerne les systèmes non-monotones existants le fragment universel de la circonscription et le fragment normal de la logique des défauts satisfont la cumulativité restreinte ainsi que la transitivité composée, et donc la conservation (dans chaque cas, la relation de conséquence sous-jacente $\vdash$ est la logique classique). Sur la question des liens entre transitivité et transitivité composée, le fragment normal de la logique de défauts est particulièrement intéressant puisqu'il respecte la transitivité composée mais pas la transitivité elle-même. Et naturellement, toute relation vérifiant la transitivité (plus la conservation relativement à une relation de conséquence) vérifie aussi la transitivité composée.

Toutes les propriétés structurelles examinées jusqu'ici caractérisent des classes de logiques indépendamment de leurs langages. Or, il y a du sens à caractériser des classes de logiques à l'aide de propriétés mentionnant des éléments de langages logiques comme des connecteurs par exemple. C'est ce qu'a fait Tarski (1930), à partir de la négation et de l'implication, pour les relations de conséquence. Dans le cas de l'implication, deux propriétés essentielles sont les suivantes:

Règle de détachement: si $\mathfrak{C} \vdash A \rightarrow B$ alors $\mathfrak{C},A \vdash B$

Principe de déduction: si $\mathfrak{C},A \vdash B$ alors $\mathfrak{C} \vdash A \rightarrow B$

La règle de détachement et le principe de déduction suggèrent immédiatement deux propriétés correspondant à la transitivité et à la transitivité composée, mais avec le connecteur d'implication pour représenter un lien d'inférence:

Modus ponens: si $\mathfrak{C} \mathrel{|\!\sim} A$ et $\mathfrak{C} \mathrel{|\!\sim} A \rightarrow B$ alors $\mathfrak{C} \mathrel{|\!\sim} B$

Modus ponens composé: si $\mathfrak{C} \mathrel{|\!\sim} A$ et $\mathfrak{C} \vdash A \rightarrow B$ alors $\mathfrak{C} \mathrel{|\!\sim} B$

La différence entre modus ponens et modus ponens composé est analogue à la différence entre transitivité et transitivité composée: il se peut qu'une relation $\mathrel{|\!\sim}$ respecte le modus ponens composé mais pas le modus ponens lui-même tandis que toute relation $\mathrel{|\!\sim}$ vérifiant le modus ponens (plus la conservation relativement à une relation de conséquence $\vdash$) vérifie aussi le modus ponens composé.

Si une relation $\mathrel{|\!\sim}$ satisfait la transitivité composée relativement à une relation de conséquence $\vdash$ vérifiant la règle de détachement alors cette relation $\mathrel{|\!\sim}$ satisfait le modus ponens composé. De manière duale, si une relation $\mathrel{|\!\sim}$ satisfait le modus ponens composé relativement à une relation de conséquence $\vdash$ vérifiant le principe de déduction alors cette relation $\mathrel{|\!\sim}$ satisfait la transitivité composée dans sa forme finitaire:

si $\mathfrak{C} \mathrel{|\!\sim} A$ et $\mathfrak{C},A \vdash B$ alors $\mathfrak{C} \mathrel{|\!\sim} B$

ou la transitivité composée dans sa forme générale, sous l'hypothèse que la relation $\vdash$ vérifie aussi la compacité descendante.

Il existe certes des relations $\mid\!\sim$ (satisfaisant la conservation relativement à une relation de conséquence $\vdash$ vérifiant à la fois le principe de déduction et la règle de détachement) qui satisfont la transitivité mais pas le modus ponens ou vice-versa. Mais la transitivité et le modus ponens sont liées ensemble de façon similaire à leurs versions composées. D'une part, si une relation $\mid\!\sim$ satisfait le modus ponens et le principe de déduction alors cette relation $\mid\!\sim$ satisfait également la transitivité dans sa forme finitaire:

si $\mathfrak{T} \mid\!\sim A$ et $\mathfrak{T}, A \mid\!\sim B$ alors $\mathfrak{T} \mid\!\sim B$

ou la transitivité dans sa forme générale, sous l'hypothèse que la relation $\mid\!\sim$ vérifie de plus la compacité descendante. D'autre part, si une relation $\mid\!\sim$ satisfait la transitivité et la règle de détachement alors cette relation $\mid\!\sim$ satisfait également le modus ponens.

Toutefois, ce dernier énoncé est vain pour les relations $\mid\!\sim$ qui sont effectivement non-monotones. La raison en est que pour une relation vérifiant le modus ponens, le fait de satisfaire la règle de détachement essentiellement entraîne la monotonie, pourvu que la relation considérée vérifie une propriété classique sur la cohérence, dite:

Ex falso: $\mathfrak{T} \mid\!\sim A \rightarrow (\neg A \rightarrow B)$

Il est possible de traiter d'autres conditions de cohérence au lieu du ex falso, et de modus ponens composé ou de transitivité (simple ou composée) au lieu du modus ponens, ce qui donne des variantes à peu près de même force que le résultat de base (Besnard 1988):

Toute relation $\mid\!\sim$ non-monotone pour laquelle il existe $\mathfrak{T}$ et A tels que $\mathfrak{T} \mid\!\sim A$ et $\mathfrak{T}, \neg A \not\mid\!\sim A$ ne peut vérifier à la fois le modus ponens, le ex falso et la règle de détachement.

Ou encore, une relation $\mid\!\sim$ qui satisfait le modus ponens composé relativement à une relation de conséquence vérifiant le ex falso, qui est non-monotone par l'existence de $\mathfrak{T}$ et A tels que $\mathfrak{T} \mid\!\sim A$ et $\mathfrak{T}, \neg A \not\mid\!\sim A$, ne peut vérifier la règle de détachement.

Dans cette formulation, à la place du ex falso, il est possible d'avoir l'axiome

$A \rightarrow (\neg A \rightarrow A)$

et il est aussi possible de considérer d'autres axiomes tels que

$A \rightarrow (B \rightarrow A)$

ou tels que, si le symbole absurde $\perp$ est disponible dans le langage,

$A \rightarrow ((A \rightarrow \perp) \rightarrow \perp)$

pourvu que l'élimination de la double négation

$$\frac{\mathfrak{T}, A \rightarrow \bot \vdash \bot}{\mathfrak{T} \vdash A}$$

soit satisfaite.

La conclusion est qu'une approche des relations non-monotones impose certaines contraintes sur le connecteur d'implication. Les résultats précédemment évoqués (Besnard 1988) (Besnard & Moinard 1994) indiquent les diverses possibilités qui s'offrent: soit affaiblir le connecteur d'implication, c'est l'approche basée sur les logiques conditionnelles (Delgrande 1988) (Nute 1988), soit rejeter le ex falso, c'est l'approche basée sur les logiques paraconsistantes (Carnielli & Lima Marques 1992) (Besnard 1990), soit les deux, c'est l'approche basée sur les logiques relevantes (Patel-Schneider 1985), ou pour prendre une autre voie, rejeter la démarche non-monotone, c'est-à-dire adopter la solution de la révision de théories (Nebel 1989).

Enfin, les relations non-monotones qui admettent le modus ponens, simple ou composé, peuvent satisfaire ou non le principe de déduction (Besnard 1988). Ainsi, pour la caractérisation de classes de relations non-monotones, la méthode axiomatique est plus souple que la méthode sémantique, à laquelle est consacrée la prochaine section.

2.2. Approche sémantique

La logique classique décrit, par l'intermédiaire des interprétations, l'ensemble de toutes les possibilités et il est donc immédiat d'en tirer les inférences qui n'admettent aucun contre-exemple. Mais pour les raisonnements conjecturels, ce qu'il convient de considérer ce ne sont pas toutes les possibilités car l'objet même d'une conjecture est de privilégier certaines d'entre elles. Dans un contexte conjecturel, certaines interprétations apparaissent comme devant être prises en compte de préférence à d'autres. D'où l'idée d'un ordre entre interprétations pour représenter ladite préférence. C'est l'approche sémantique, par les modèles préférentiels (Besnard & Siegel 1986) (Shoham 1987), de la caractérisation de classes de systèmes non-monotones. Une relation (qui est en général un ordre partiel strict) $\subset$ est donc donnée entre les interprétations, avec $\mathfrak{N} \subset \mathfrak{M}$ signifiant que $\mathfrak{N}$ est préféré à $\mathfrak{M}$. Par ailleurs, $\mathfrak{M}$ est un modèle préférentiel de $\mathfrak{U}$ s'il n'existe pas de modèle $\mathfrak{N}$ de $\mathfrak{U}$ tel que $\mathfrak{N} \subset \mathfrak{M}$ (un modèle préférentiel est donc un modèle minimal pour $\subset$). Une notion d'inférence préférentiel $\models_{\subset}$ en est extraite, où $\mathfrak{T} \models_{\subset} A$ si tous les modèles préférentiels de $\mathfrak{T}$ sont aussi des modèles de A.

Avec les modèles préférentiels, il devient possible de redéfinir des systèmes non-monotones dans un cadre unifié (Besnard & Siegel 1986): par exemple, en prenant comme relation $\subset$ un préordre (basé sur une partition <R=, R+, R-, R*> des symboles de prédicats) tel que $\mathfrak{N} \subset \mathfrak{M}$ si $\mathfrak{N}$ et $\mathfrak{M}$ sont complètement identiques si ce n'est que:

- pour tout symbole de prédicat P dans R+, la valeur associée à P par $\mathfrak{M}$ est une sous-relation de la valeur associée à P par $\mathfrak{N}$
- pour tout symbole de prédicat P dans R-, la valeur associée à P par $\mathfrak{N}$ est une sous-relation de la valeur associée à P par $\mathfrak{M}$
- pour tout symbole de prédicat P dans R*, les valeurs associées à P par $\mathfrak{N}$ et $\mathfrak{M}$ sont quelconques.

Par contrecoup, pour tout symbole de prédicat P dans R=, $\mathfrak{M}$ et $\mathfrak{N}$ doivent associer à P exactement la même valeur (qui est une relation).

En illustration de la possibilité de redéfinir des systèmes non-monotones selon l'approche qui vient juste d'être exposée, la circonscription d'un prédicat P avec les prédicats $P_1,\ldots,P_n$ pouvant varier, correspond à la relation $\models_{\subset}$ où $\subset$ est le préordre basé sur la partition <R=, { }, {P}, $\{P_1,\ldots,P_n\}$> dans laquelle R= est l'ensemble de tous les symboles de prédicats en dehors de $\{P, P_1,\ldots,P_n\}$.

L'approche par les modèles préférentiels est assez générale car des interprétations modales par exemple peuvent être considérées au lieu des interprétations de la logique classique (Shoham 1987).

Il est souhaitable que la relation $\subset$ possède une propriété d'existence de minorant. En effet, étant donné un ensemble de prémisses $\mathfrak{T}$, si pour tout modèle non préférentiel $\mathfrak{M}$ de $\mathfrak{T}$ il existe un modèle préférentiel $\mathfrak{N}$ de $\mathfrak{T}$ tel que $\mathfrak{N} \subset \mathfrak{M}$ alors la relation $\models_{\subset}$ est incohérente sur $\mathfrak{T}$ ssi la relation $\models$ est incohérente sur $\mathfrak{T}$. Autrement, étant entendu que $\mathfrak{T} \not\models$ A pour au moins une formule A, si par exemple pour tout modèle $\mathfrak{M}$ de $\mathfrak{T}$ il existe un modèle $\mathfrak{N}$ de $\mathfrak{T}$ tel que $\mathfrak{N} \subset \mathfrak{M}$ alors $\mathfrak{T} \models_{\subset}$ B pour toutes les formules B. En d'autres termes, s'il n'existe pas de minorant, relativement à $\subset$, sur les modèles de prémisses conjointement cohérentes au sens de $\models$, rien ne garantit que $\models_{\subset}$ préserve la cohérence de ces prémisses (Bossu & Siegel 1985).

La propriété d'existence de minorant est une condition élégante qui n'a pas de correspondant dans l'approche axiomatique, où il faut explicitement considérer

Cohérence relative: si $\mathfrak{T} \mid\!\sim \bot$ alors $\mathfrak{T} \vdash \bot$

Evidemment, l'approche sémantique offre l'intérét d'obéir nécessairement à la contrainte dite "conservation" dans la section précédente. C'est-à-dire qu'un système non-monotone ainsi défini est toujours une extension d'une véritable relation de conséquence. Dans l'approche axiomatique, la recherche d'une relation de conséquence $\vdash$ qui soit la fondation d'un système non-monotone arbitraire est moins évidente.

Si la relation $\mid\!\sim$ est réflexive, il est possible d'en définir une sous-relation $\vdash_0$ par

$\mathfrak{T} \vdash_0$ A ssi A $\in \mathfrak{T}$

Puisque $\vdash_0$ satisfait trivialement la coupure, c'est la plus petite relation de conséquence pour laquelle $\mid\!\sim$ respecte la conservation.

Une autre manière simple de définir une sous-relation monotone d'un système $\mid\!\sim$ arbitraire est la suivante:

$\mathfrak{T} \vdash_1 A$ ssi pour tout $\mathfrak{T}'$, si $\mathfrak{T} \subseteq \mathfrak{T}'$ alors $\mathfrak{T}' \mathrel{|\!\sim} A$

Il est aisé de vérifier que $\vdash_1$, lorsqu'elle est une relation de conséquence, est la plus grande qui soit sous-jacente à $\mathrel{|\!\sim}$. En effet, il est clair que si $\mathrel{|\!\sim}$ est réflexive alors $\vdash_1$ l'est aussi. De plus, si $\mathrel{|\!\sim}$ satisfait la transitivité (même restreinte au cas où $\mathfrak{A}$ est une formule) alors $\vdash_1$ satisfait la coupure.

Bien entendu, il y a de nombreuses possibilités.

Au contraire de l'approche axiomatique, l'approche par les modèles préférentiels est très contrainte par les connecteurs. Premièrement, pour toute logique qui se définit par la méthode des modèles préférentiels, la règle de détachement est équivalente à la version finitaire de la monotonie (Shoham 1987). En effet, tout relation $\models_{\subset}$ vérifie

si $\mathfrak{T} \models_{\subset} A \to B$ alors $\mathfrak{T}, A \models_{\subset} B$

ssi elle vérifie

si $\mathfrak{T} \models_{\subset} A$ alors $\mathfrak{T}, B \models_{\subset} A$

Deuxièmement, le principe de déduction est vérifié par toute logique qui se définit par la méthode des modèles préférentiels (Shoham 1987)

si $\mathfrak{T}, A \models_{\subset} B$ alors $\mathfrak{T} \models_{\subset} A \to B$

Ceci fixe une limite (Besnard 1988) à la généralité de cette approche sémantique de la caractérisation de classes de relations non-monotones car ne peuvent alors être pris en compte des logiques où les inférences doivent pouvoir se développer à l'intérieur d'un certain système de ressources bornées (ce sont notamment toutes les logiques rejetant le principe d'omniscience, lequel stipule que connaître par exemple les règles d'un jeu équivaut à tout connaître sur ce jeu -abstraction faite bien sûr de ce qui est subjectif, comme les aspects psychologiques du jeu).

*IRISA-INRIA, Rennes

BIBLIOGRAPHIE

Besnard, Ph.: 1988, 'Axiomatizations in the metatheory of non-monotonic inference systems', in *Proc. 7th Conf. CSCSI*, Edmonton, Canada, 117-124.

Besnard, Ph. & Moinard, Y.: 1994, 'Implication connectives for logics with right weakening', in *Proc. IPMU-94*, Paris, France, 694-699.

Bossu, G. & Siegel, P.: 1985, 'Saturation, nonmonotonic reasoning and the closed-world assumption', *Artificial Intelligence* 25, 13-63.

Carnielli, W.A. & Lima Marques, M.: 1992, 'Reasoning under inconsistent knowledge', *Journal of Applied Non-Classical Logics* 2, 49-79.

Clark, K.: 1978, 'Negation as failure', in Gallaire H. & Minker J. (eds.): *Logic and databases*, New York, Plenum Press.

Delgrande, J.: 1988, 'An approach to default reasoning based on a first order conditional logic', *Artificial Intelligence* 36, 63-90.

Gabbay, D.M.: 1985, 'Theoretical foundations for non-monotonic reasoning in expert systems, in Apt K.R. (ed.): *Logics and Models of Concurrent Systems*, Berlin, Springer-Verlag.

Gentzen, G.: 1955, *Recherches sur la déduction logique*. Trad. franc. de 'Untersuchungen über das logische Schliessen' (*Mathematische Zeitschrift* 39, 1933, 176-210, 405-431), Paris, Presses Universitaires de France.

Kraus, S., Lehmann, D. & Magidor, M.: 1990, 'Nonmonotonic reasoning, preferential models and cumulative logics', *Artificial Intelligence* 44, 167-207.

Makinson, D.: 1989, 'General theory of cumulativity', in Reinfrank, M., de Kleer, J., Ginsberg, M.L. & Sandewall, E. (eds.): *Proc. 2nd Int. Workshop on Non-Monotonic Reasoning* (Grassau, Germany), LNCS 346, Berlin, Springer-Verlag.

Montague, R.: 1963, 'Syntactical treatment of modality; with corollaries on reflection principles and finite axiomatizability', *Acta Philosophica Fennica* 16, 153-167, réimprimé dans *Formal philosophy: selected papers of Richard Montague*, Montague, R., Yale University Press, 1974.

Nebel, B.: 1989, 'A knowledge level analysis of belief revision', in *Proc. Conf. on principles of knowledge representation and reasoning*, Toronto, Canada, 301-311.

Nute, D.: 1988, 'Defeasible reasoning and decision support systems', *Decision Support Systems* 4, 97-110.

Patel-Schneider, P.F.: 1985, 'A decidable first order logic for knowledge representation', in *Proc. IJCAI-85*, Los Angeles, 455-458.

Shoham, Y.: 1987, 'Non-monotonic logics: meaning and utility', in *Proc. IJCAI-87*, Milano, Italy, 388-393.

Tarski, A.: 1972, 'Sur quelques concepts fondamentaux de la métamathématique'. Trad. franc. de 'Über einige fundamentale Begriffe der Metamathematik', *C. R. Soc. Sciences et Lettres de Varsovie*, vol. 23, 1930, cl. III, 22-29, parue dans *Logique, sémantique, métamathématique*, tome 1, Tarski, A., Paris, Armand Colin.

Tarski, A.: 1939, 'On undecidable statements in enlarged systems of logic and the concept of truth', *Journal of Symbolic Logic*, vol. 4, 105-112.

A GENERAL THEORY OF STRUCTURED CONSEQUENCE RELATIONS[1]

Dov M. GABBAY*

ABSTRACT

There are several areas in logic where the monotonicity of the consequence relation fails to hold. Roughly these are the traditional non-monotonic systems arising in Artificial Intelligence (such as defeasible logics, circumscription, defaults, etc), numerical non-monotonic systems (probabilistic systems, fuzzy logics, belief functions), resource logics (also called substructural logics such as relevance logic, linear logic, Lambek calculus), and the logic of theory change (also called belief revision, see Alchourron, Gärdenfors, Makinson [22-24]). We are seeking a common axiomatic and semantical approach to the notion of consequence which can be specialised to any of the above areas. This paper introduces the notions of structured consequence relation, shift operators and structural connectives, and shows an intrinsic connection between the above areas.

1. Introduction

We want to give an answer to the following question: What is a logical system? Obviously our starting point should be a language which allows us to express declarative statements and we need to give a formal definition of when we accept a reasoning system based on the language as a logical system.

The traditional answer to that (say in 1975) was that to present a logical system we need to define the well formed formulas of the language, say **L**, and further define a consequence relation for **L**. This is a relation between finite sets of formulas Δ, Γ of **L**, written $\Delta \vdash \Gamma$, satisfying the properties of Reflexivity, Monotonicity and Cut.

Reflexivity: $\Delta \vdash \Gamma$ if $\Delta \cap \Gamma \neq \emptyset$
Monotonicity: $\Delta \vdash \Gamma$ implies $\Delta, \Delta' \vdash \Gamma, \Gamma'$
Cut[2]: $\Delta \vdash A$; $\Delta, A \vdash \Gamma$ imply $\Delta \vdash \Gamma$

The consequence relation may be defined in various ways. Either through an algorithmic system $\mathbf{S}_{\vdash}$, or implicitly by postulates on the properties of $\vdash$.

Thus a logic is obtained by specifying **L** and $\vdash$. Two algorithmic systems $\mathbf{S}_1$ and $\mathbf{S}_2$ which give rise to the same $\vdash$ are considered the same logic.

It soon became apparent, through the applications of logic in the analysis of language and in Artificial Intelligence, that there are many reasoning systems

THEORIA - Segunda Época - Vol. X
1995, Nº 23, 49-78

which we would intuitively like to admit into the family of logics, which are not covered by the above definition. These include a variety of non-monotonic systems, families of resource logics and a large class of probabilistic and network systems.

These reasoning systems are mostly presented algorithmically, as proof systems **S**, and the majority of them give rise to a consequence relation which does not satisfy the postulates above. Further, there are many different proof systems for the *same* consequence relation, which are of such different motivation, mathematical presentation and intuitions, that one does not feel it is correct to identify them as the *same* logical systems.

It was obvious that we needed to modify our concept of a logical system in response to current reasoning practices.

The first attempt in this direction was in 1985, in [1], where we proposed that a consequence relation should only be required to satisfy *Restricted Monotonicity* and not full monotonicity, namely:

Restricted Monotonicity: $\Delta \vdash A, \Delta \vdash \Gamma$ imply $\Delta, A \vdash \Gamma$.

The above weakening on the conditions of the notions of a consequence relation allows one to accept more reasoning systems as logical systems.

This idea was further developed in the literature by several authors, thus creating the new area of *axiomatic non-monotonic reasoning.*

Makinson [2] proposed a semantics for the above basic logic (i.e. the smallest consequence relation satisfying the above three conditions) and proved a completeness result. Kraus, Lehmann and Magidor [3] studied a variety of extensions of the basic system, providing them with preferential semantics in the spirit of Shoham [4, 5]. Several other authors continued to develop this area, among them Wojcicki [6, 7], Lehmann [8], Lehmann and Magidor [9], Besnard [10] and Akama [11]. A general survey is available in Makinson [12].

The above notion of consequence, although general enough to cover many non-monotonic systems such as circumscription and negation by failure, does exclude, unfortunately, such non-monotonic systems as probabilistic systems, resource logics such as linear logic and other such systems such as inheritance networks. Obviously a more general notion is required, a notion which can unify the probabilistic approach with the rule based approach; to name one important need.

On the algorithmic front, we have no alternative but to accept that different proof systems should be considered as different logics, even though they give rise to the same consequence relation. This view was put forward in 1983, and supported at length in [13,16 and 31].

Thus the situation in 1988-89 was that although we were able to accept more systems into the family of logics we still had to exclude many well known reasoning systems.

Furthermore, there was no unifying framework and a satisfactory general answer to what is a logical system. In 1989, we turned to proof theory [13] and introduced the notion of *Labelled Deductive Systems (LDS).*

This is a systematic framework where the logical units are labelled formulas (of the form t:A, where t is a term in some algebra) and where the logical rules are rules for manipulating both formulas and their labels. The intuition behind *LDS* is that in t: A, A carries declarative information and the label t, represents further information of a different nature which we do not want to "code" into A. It turns out that many monotonic, non-monotonic and probabilistic systems can be presented as *LDS* systems. An *LDS* database Δ is a constellation of labelled formulas with additional structure on the labels. For example Δ may be an algebra τ with a function $\delta(x)$, $x \in \tau$, assigning formulas $A_x = \delta(x)$, for each $x \in \tau$.

Proof rules are given to define the notion $\Delta \vdash$ s: B. This approach is developed in detail in [13]. It is possible to have $\Delta \vdash s_i$: B for several labels s_i, for example s_i may show from which part of Δ the formula B is proved. In many applications one is interested in the notion of $\Delta \vdash$ B, where we want to ignore the label and just give a yes or no answer to the query "Does B follow from Δ". This can be done in several ways. The process of extracting (through some proof method) a decision of the form $\Delta \vdash$ B or $\Delta \vdash \sim$B or neither, from the set $\{t \mid \Delta \vdash t: B\}$ { $\{t \mid \Delta \vdash t{:}\sim B\}$ is called *Flattening*. Obviously the flattening process is functionally dependent on the structure of Δ (as induced by the labels). The flattening process is actually *hiding* the labels and all it really uses is a structured database Δ proving or not proving a formula B. We can in this case define axiomatically the properties of the consequence $\Delta \vdash$ B. We thus end up with a notion of consequence between Δ and B, where Δ is structured and the labels in Δ are hidden, and used only in the proof system. It turns out that this simplified notion is very useful and can be developed on its own as a direct extension of the notion of monotonic consequence relation. We call our studies the area of *structured consequence relations*. It can be developed independently of *LDS*, as a direct contribution to the abstract area of *axiomatic non-monotonic reasoning.*

The purpose of this paper is to develop the notion of structured consequence relation and further refine the notion of a non-monotonic consequence relation to relate to non monotonic systems arising from resource boundedness, and controlled inference. The refinement is in the direction of structuring the assumptions. Some pioneering work in the direction of looking at structural connectives can already be found in [19].

Many non-monotonic databases are naturally structured. The database may be, for example, a list of assumptions, or a partially ordered set of assumptions, and consequences are proved from the database using an inferential (logical) discipline which takes into account such a structure. Non-monotonicity may arise because of these "resource" considerations and inferential restrictions.

We begin our study with two motivating examples.

Example 1. (Concatenation Logic). A database in this logic is a sequence of formulas $(A_1,\ldots,A_n)$. The inference rule is modus ponens. To derive B from the database we must use all the assumptions in the order shown. Further, when we use modus ponens A, A $\rightarrow$ B $\vdash$ B, the implication A $\rightarrow$ B must be supported by assumptions earlier in the database sequence than all those which support the minor premise A. Thus, for example

(a_1) $A \rightarrow (B \rightarrow C)$

(a_2) B
(a_3) A

does not prove C because we have to start by using (a_1) and (a_3)"jumping over" (a_2). However,

(b_1) $A \rightarrow (B \rightarrow C)$
(b_2) A
(b_3) B

does prove C because (b_1) and (b_2) give ($B \rightarrow C$) then (b_3) is used and we get C.

Another example is:

(c_1) $A \rightarrow (A \rightarrow B)$
(c_2) A

This database does not prove B because (c_2) needs to be used twice. However, the database:

(d_1) $A \rightarrow (A \rightarrow B)$
(d_2) A
(d_3) A

does prove B.

Another example is:

(e_1) $A \rightarrow (A \rightarrow B)$
(e_2) A
(e_3) A
(e_4) C

This database does not prove B because (e_4) is not used. Even if (e_4) were B (i.e. C = B) we still could not derive B because if we use (e_4) we are not using all the assumptions and we are not starting our proof at the first one. The alternative, using (e_1)-(e_3) is also not valid because (e_4) is not used.

Concatenation logic has applications in any area where the database is perceived as a list, either for conceptual reasons or for computational reasons.

As an example we consider its applications to Prolog. Consider the database

1. $q \rightarrow q$
2. q

and the query ?q.

Prolog implementations will store the database as a list and compute the query from the top of the list. This query will therefore loop. In concatenation logic, the body of clause 1 needs to be asked from clause 2. Thus concatenation logic gives a discipline for the Prolog pointer.

Example 2. (Defeasible Logic). This important approach to non-monotonic reasoning was introduced by Nute [15]. The idea is that rules can prove either an atom q or its negation ¬q. If two rules are in conflict, one proving q and one proving ¬q, the deduction that is stronger is from a rule whose antecedent is logically more specific. Thus the database:

Bird(x) → Fly(x)
Big(x) ∧ Bird(x) → ¬Fly(x)
Big(a)
Bird(a)

will entail ¬Fly(a) because the second rule is more specific.

We can view the above as a structured database Δ in which the ordering of the rules is done by the logical strength of their antecedents relative to some background theory Θ (which can be a subtheory of Δ of some form). Deduction pays attention to the strength of rules.

We now introduce the new notion of a non-monotonic consequence relation, which we call S-consequence relation (Structured Consequence Relation). A precise definition will be given in a later section. We need three auxiliary notions for our definition:

1. We have already indicated that we need to deal with structured databases. We thus have to specify precisely what structures are allowed as databases, for the consequence relation to be defined. These must include the one formula database (A) and the empty database Ø.

2. The notion of a structured database must also include the concept of *structured addition of data.* By this we mean the notion of how to combine two databases together. The traditional way is to take their union but when the databases are already structured, the union has to be structured as well. We denote the structured addition of the databases Δ and Γ by $\Delta+\Gamma$. This binary operation, "+", has to be defined as part of the concepts underlying the consequence relation. It usually satisfies associativity

$$\Delta_1 + (\Delta_1 + \Delta_3) = (\Delta_1 + \Delta_2) + \Delta_2$$

and the empty database must satisfy

$$\varnothing + \Delta = \Delta + \varnothing = \Delta$$

Commutativity is not required to hold.

3. We also need a concept of substitution of one structured database Δ inside another, Γ, achieved by replacing one formula A in Γ by Δ. Formally we need to make sense of the symbol $\Gamma[A]$, reading Γ is a structured database which contains A somewhere inside, and the symbol $\Gamma[\Delta]$, which denotes the database resulting from substituting Δ for A inside Γ. These notions are needed to formulate the Cut Rule.

We can now define the three rules which we call **Identity, Surgical Cut** (or **Substitutional Cut**) and **Directional Monotonicity** as follows:

Identity $(A) \mathrel{|\!\sim} A$

Surgical (Substitutional) Cut

$$\frac{\begin{array}{c}\Delta \mathrel{|\!\sim} A \\ \Gamma[A] \mathrel{|\!\sim} B\end{array}}{\Gamma[\Delta] \mathrel{|\!\sim} B}$$

Directional Monotonicity (right hand side)

$$\frac{\Delta \mathrel{|\!\sim} A}{\Delta + \Gamma \mathrel{|\!\sim} A}$$

Directional Monotonicity (left hand side)

$$\frac{\Delta \mathrel{|\!\sim} A}{\Gamma + \Delta \mathrel{|\!\sim} A}$$

Definition 1. (Tentative Definition of Structured Consequence Relation) Let **L** be a language. Let $\mathcal{L}$ be a family of order types of the form $(\tau, \leq)$. Assume that the empty set and one element set types are in $\mathcal{L}$. Assume that the notion of substitution of one order type τ for a point x in another order type $\tau'(x)$ is well defined (as in (3) above).

We say that a relation $\Delta \mathrel{|\!\sim} A$, (between ordered multisets Δ of wffs of an order type τ and a single wff A) is a **S-consequence relation** iff it satisfies **Identity** and **Surgical Cut**.

Example 3. (Consequence presentation of CL) Consider a language with $\rightarrow$. Consider data structures in the form of lists $\Delta = (A_1,\dots,A_n)$. An S-consequence relation on pairs (Δ, A), written as $\Delta \Vdash A$, is any relation satisfying the following two conditions.

Identity

$(A) \Vdash A$

Surgical (Substitutional) Cut for (the data structure) Lists

$$\frac{\begin{array}{l}(A_1,\dots,A_n) \Vdash A \\ (C_1,\dots,C_m, A, D_1,\dots,D_k) \Vdash B\end{array}}{(C_1,\dots,C_m, A_1,\dots,A_n, D_1,\dots,D_k) \Vdash B}$$

Let $\Vdash_{CL}$ be the smallest S-consequence relation satisfying the *right hand side deduction theorem* for the data structure namely:

Right-hand side Deduction Theorem:

$(A_1,\dots,A_n) \Vdash A \rightarrow B$ iff $(A_1,\dots,A_n,A) \Vdash B$

Note that we need to show that such a smallest $\Vdash$ exists. This is simple because the family of S-consequence relations for lists which satisfy **Identity, Surgical Cut** and **Right hand side Deduction Theorem** is closed under intersections.

The above example illustrates two points:

1. How to define a logic by conditions on its consequence relation and its data structures.
2. The notions of *a* Cut Rule and*a* Deduction Theorem depend on the data structures.

To further illustrate our ideas, note that the well-known intuitionistic logic can be defined in a similar manner, as shown next.

1. *Intuitionistic Consequence*

Data structures are sets of wffs. $\Delta \Vdash A$ is the smallest consequence relation on the data structure satisfying Reflexivity, Monotonicity, Cut and the Deduction Theorem.

2. *R-mingle Consequence*

Data structures are sets of wffs. $\Delta \Vdash A$ is the smallest consequence relation satisfying Identity, Cut and the Deduction Theorem.

3. *Linear Consequence* (**LL**)

Data structures are multisets of formulas. $\Delta \Vdash A$ is the smallest Consequence Relation satisfying Identity, Cut and the Deduction Theorem.

Notice that the difference between R-mingle and linear consequence is only in the data structures.

2. Structured Consequence Relations: A General Notion of a Logical System

We have seen that the general notion of a database is a configuration of formulas. The proof discipline from such databases would rely on the configuration in defining the allowable proof moves.

This section develops the general notions needed for S-consequence relation.

Definition 2. 1. Let **L** be a language for well formed formulas. **L** will contain atomic propositions and connectives and the notion of a wff is defined inductively in the traditional way.

2. Let $\mathcal{L}$ be a theory in first or higher order logic and let $\mathcal{M}$ be a class of classical models of $\mathcal{L}$. We assume that the structure with one point and empty relations is in $\mathcal{M}$. This structure is denoted by «t». Assume that two model theoretic operations are defined on $\mathcal{M}$ and assume that $\mathcal{M}$ is closed under these operations. The operations are the following:

(a) If τ_1 and τ_2 are two structures in $\mathcal{M}$ then an operation + is available for constructing the structure $\tau = \tau_1 + \tau_2$. "+" usually satisfies associativity, but not necessarily in the general case. If $\varnothing$ is the empty set then + is defined and $\varnothing + \tau = \tau + \varnothing = \tau$.

(b) Let τ_1 and τ_2 be two disjoint structures in $\mathcal{M}$ and let $x \in \tau_1$ be a point in τ_1. Then a binary substitution function $Sub\, \tau_2^x\, \tau_1(x)$ is defined which yields the result of substituting τ_2 in $\tau_1(x)$ for x (i.e. $\lambda x \tau_1(x)[\tau_2]$ is well defined). The result of the substitution is also in $\mathcal{M}$. *Sub* is associative, and behaves like substitution if done properly (no clash of variables).

(c) For some purposes (see section on structural connectives) we need to allow for the notion of *deletion*. That is we must know how to delete the point x from $\tau(x)$. We can regard deletion as substituting the empty structure $\varnothing$ for x, i.e. $\lambda x \tau(x)[\varnothing]$.

Note that (a)-(c) above are very general, almost saying nothing. For each particular case, '+' and '*Sub*' must be specifically defined. For our purpose, i.e. the study of general structured consequence relations, we do not need to know more. See [35] for a general algebraic example with a semantical interpretation.

3. A database is any pair (τ, δ) where $\tau \in \mathcal{M}$ and δ is a function assigning for each $x \in \tau$ a wff $\delta(x)$.

4. A consequence relation for the pair $(\mathcal{M}, \mathbf{L})$ is any relation $\mid\!\sim$ between databases $\Delta = (\tau, \delta)$ and single wffs A, written in the form $\Delta \mid\!\sim A$, satisfying the following:

(a) **Identity**

(A) $\mid\!\sim$ A

(b) **Surgical Cut for $\mathcal{M}$.** If

$$\Delta = (\tau_1, \delta_1),\ \Gamma = (\tau_2, \delta_2),\ \tau_1 \cap \tau_2 = \varnothing,\ x \in \tau_2 \text{ and } \delta_2(x) = A$$

and if further

$$\Delta \mid\!\sim A$$
$$\Gamma \mid\!\sim B$$

then

$$\Gamma[\Delta] = (\tau_3, \delta_3) \mid\!\sim B$$

where

$$\tau_3 = Sub\, \tau_1^x\, \tau_2(x)$$

and

$$\delta_3(y) = \begin{cases} \delta_1(y) \text{ if } y \in \tau_1 \\ \delta_1(y) \text{ if } y \in \tau_2 \end{cases}$$

Example 4. Consider **CL** (or examples 1 and 3) and its data structures of lists of wffs. Let $\Delta = (A_1,\ldots,A_n)$ and consider A of examples 1 and 3. This new data item can be "added" to Δ in several ways:

1. Add A to the end of the sequence to form
   $(A_1,\ldots,A_n, A)$
2. Add A to the beginning of the sequence to form
   $(A, A_1,\ldots,A_n)$
3. Add A to the middle of the sequence to form
   $(A_1,\ldots,A_k, A, A_{k+1},\ldots,A_n)$
4. Of course we can have combinations, for example, add A to the middle or the beginning, etc

The **CL** deduction theorem allows for A to be added to the end of the sequence. We can thus write $A \rightarrow_e B$ for the **CL** implication to indicate that given A it can give B provided A is added to the end of the data sequence. The next connective to consider is the implication $A \rightarrow_b B$ where "*b*" means "beginning". Here we expect a deduction theorem of the form

$$(A, A_1,\ldots,A_n) \vdash B,$$

iff

$$(A_1,\ldots,A_n) \vdash A \rightarrow_b B$$

For example, since

$$A \rightarrow_b B \vdash A \rightarrow_b B$$

we will get in this logic

$$A, A \rightarrow_b B \vdash B.$$

In concatenation logic the implication must come earlier than the minor premise. In this logic, the minor premise comes first. The set of theorems of this logic, with $\rightarrow_b$, is unchanged; we get **CL** again, except that in the semantics we now concatenate to the left. A different system is obtained if we have both implications (see [17]). This is actually the Lambek Calculus. The Lambek Calculus can be defined as the smallest S-consequence relation with two implications $\rightarrow_e$ and $\rightarrow_b$ satisfying the appropriate Deduction Theorems for the list data structures.
For example, since

$$A \rightarrow_e B, A \vdash B$$

we get that

$$A \vdash (A \rightarrow_e B) \rightarrow_b B$$

The stipulation that

$A \rightarrow_e B$ iff $A \rightarrow_b B$

gives us linear logic, with $\rightarrow$ being either of $\rightarrow_b$ or $\rightarrow_e$.

Both $\rightarrow_e$ and $\rightarrow_b$ satisfy a Deduction Theorem.

Having illustrated our concepts, we can now generalise the notion of the deduction theorem to arbitrary data constellations. To achieve that we also need to generalise the notion of implication.

Definition 3. 1. Let $\mathcal{M} = \{\tau_i\}$ be a class of structures τ_i, as in definition 1. Let ϕ be a wff in some language $\mathcal{L}$ capable of expressing properties of the structures in $\mathcal{M}$

Assume that for every τ in $\mathcal{M}$, ϕ defines a unique element (or possibly a unique set of elements) in τ.

The formula ϕ defines a contraction mapping on structures τ. Let τ' be the structure obtained from τ by taking out the points identified by ϕ. Thus, for example, for lists structures τ, if ϕ identifies the last element in the list then τ' is the list without its last element. We can denote the contraction mapping function by "$\Vert\phi$" and write $\tau' = \tau \Vert \phi$.

Thus, given two structures τ_1, τ_2 we may say that $\tau_2 = \tau_1 \Vert \phi$ iff τ_2 is obtained by dropping out of τ_1 the elements identified by ϕ. We write $\tau\,[\Vert\phi]^n$ to mean $\tau \Vert \phi,\ldots,\Vert\phi$, n times. For example, for lists τ and ϕ which identifies the last element of the list, we have $(t_1,\ldots,t_n) = (t_1,\ldots,t_n, x) \Vert \phi$ and $\varnothing = (t_1,\ldots,t_n)\,[\Vert\phi]^n$.

2. Let a database discipline be given, involving a family of structures $\mathcal{M}$. A database Δ in this discipline is any pair (τ, δ) where τ is a finite element of $\mathcal{M}$ (e.g. a list) and δ associates with each $t \in \tau$ a formula $\delta(t) = At$. We also write $\Delta = \Delta' \Vert \phi$, just in case $\Delta = (\tau, \delta)$, $\Delta' = (\tau', \delta')$ and $\tau = \tau' \Vert \phi$ and $\delta = \delta' \upharpoonright \tau$, i.e. δ is the restriction of δ' to τ.

Example 5. Let $\mathcal{M}$ be the class of all finite lists including the empty list. Define + as concatenation. Let $\mathcal{L}$ be a language with "+" and "<". ϕ can be a formula in $\mathcal{L}$ which defines uniquely the last element of any finite list. Thus for $\tau = (a_1,\ldots,a_n, b)$, we have $\tau \Vert \phi = (a_1,\ldots, a_n)$. These lists are the data structures of **CL**.

Definition 4. (The Notion of the Deduction Theorem) Let Δ, Δ' be two databases and assume that $\tau = \tau' \Vert \phi$. Assume further that we have

$$\Delta \Vert \phi \mathrel{|\!\sim} A \rightarrow B \text{ iff } \Delta' \mathrel{|\!\sim} B$$

Consider a special connective denoted by $\rightarrow_\phi$. We say that $\rightarrow_\phi$ satisfies the *Deduction Theorem with respect to* ϕ iff whenever $\Delta \mathrel{|\!\sim} B$ and $\Delta = (\tau, \delta)$ and τ has n elements then

$$\Delta[\Vert\phi]^n \mathrel{|\!\sim} \varnothing: \delta(t_1) \rightarrow_\phi (\ldots \rightarrow_\phi (\delta(t_n) \rightarrow_\phi B) \ldots)$$

where «t_i» $= \tau[\Vert\phi]^i - \tau[\Vert\phi]^{i+1}$ i.e. t_i is the element identified by ϕ in $\tau[\Vert\phi]^i$.

Our discussions so far presented the consequence relation in the form of $\Delta \mathrel{|\!\sim} A$, where A is a single formula on the right hand side. This is a Tarski type consequence relation. The general form of the consequence relation should be $\Delta \mathrel{|\!\sim} \Gamma$, where both Δ and Γ are structured databases. This is a Scott type consequence relation. In fact, there is no reason at all to assume that the same structures can serve on both sides of the turnstile. If we refer to the left hand side as *the data* and the right hand side as the goal, then the general Scott type S-consequence relation has the form $\Delta \mathrel{|\!\sim} \Gamma$, where Δ is a data structure and Γ is a goal structure. For example, Δ may be a set of formulas while Γ may be a list of formulas. This is indeed the case, for example, in Logic Programming, where we have

$\Delta \mathrel{|\!\sim} (A_1,\ldots,A_n)$ iff $\Delta \mathrel{|\!\sim} A_1$ and ... and $\Delta \mathrel{|\!\sim} A_n$

If Δ is a sequence, as in the case of **CL**, or a multiset, as in the case of linear logic, then we can have

$\Delta \mathrel{|\!\sim} (A_1,\ldots,A_n)$
iff for some Δ_i, $\Delta = \Delta_1 + \ldots + \Delta_n$ and $\Delta_i \mathrel{|\!\sim} A_i$, $i = 1,\ldots,n$

where + is concatenation or multiset union, respectively.

Another example is the case of Gentzen systems for classical logic, where both Δ and Γ are sets and where we have $\Delta \mathrel{|\!\sim} \{A_1,\ldots,A_n\}$ iff $\Delta \mathrel{|\!\sim} A_1 \vee \ldots \vee A_n$.

In the above cases the Scott consequence relation $\Delta \mathrel{|\!\sim} \Gamma$, for Γ a complex structure, can be reduced to its Tarski part $\Delta \mathrel{|\!\sim} A$, where A a single formula. It is not clear whether we should insist on this property to hold in the general case.

To get an idea of our options, let our starting point be two structured Tarski type consequence relations $\mathrel{|\!\sim}_1$ and $\mathrel{|\!\sim}_2$ on the same language **L** with $\mathcal{L}_1$, $\mathcal{L}_2$ and $\mathcal{M}_1$ and $\mathcal{M}_2$ the two structural languages satisfying the conditions of definition 2. We want to "extend" $\mathrel{|\!\sim}_1$ and $\mathrel{|\!\sim}_2$ to a Scott type consequence relation $\mathrel{||\!\sim}$, allowing for $\Delta \mathrel{||\!\sim} \Gamma$ where Δ is an $\mathcal{M}_1$ structured database and Γ is an $\mathcal{M}_2$ structured goal. Thus the allowable structures for $\mathrel{||\!\sim}$ are $\mathcal{M}_1$ for the data and $\mathcal{M}_2$ for goals. Clearly it must satisfy the rules of **Identity** and **Surgical Cut** in the following form:

$A \mathrel{||\!\sim} A$

and

$$\frac{\Delta[A] \mathrel{||\!\sim} \Gamma, \Delta' \mathrel{||\!\sim} A}{\Delta[\Delta'] \mathrel{||\!\sim} \Gamma}$$

We may wish to rename the Cut Rule as **Left hand side Surgical Cut**, to stress that cut is done on the left.

A **Right hand side Surgical Cut** would be

$$\frac{\Delta \mathrel{||\!\sim} \Delta[B] \qquad B \mathrel{||\!\sim} \Gamma'}{\Delta \mathrel{||\!\sim} \Gamma[\Gamma']}$$

The question is, what other requirements do we have for $\mathrel{||\!\sim}$?

We obviously want to make use of both $\mid\!\sim_1$ and $\mid\!\sim_2$. The new consequence $\Vdash\!\sim$ should extend them both in some sense. Since both consequence relations are general, the only structures they share for certain are the unit formulae structures A. Given Δ, Γ and A, we know what $\Delta \mid\!\sim_1 A$ is and what $\Gamma \mid\!\sim_2 A$ is, but no more.

The next two definitions tell us what is the general notion of a Scott type S-consequence relation and how to form a Scott type $\Vdash\!\sim$ out of $\mid\!\sim_1$ and $\mid\!\sim_2$.

Definition 5. Let **L** be a language for wffs and let $(\mathcal{M}_1, \mathcal{L}_1)$ and $(\mathcal{M}_2, \mathcal{L}_2)$ be two languages for data and for goal structures respectively. Note that $\mathcal{L}_1$ and $\mathcal{L}_2$ come with their own respective notions of structural substitutions and structural additions $+_1$ and $+_2$ respectively.

A relation $\Delta \Vdash\!\sim \Gamma$ is said to be a (Scott type) S-consequence relation on the data and goal structures iff it satisfies the following two conditions:

Identity $A \Vdash\!\sim A$

Surgical Cut

$$\frac{\Delta[A] \Vdash\!\sim \Gamma[B] \quad \Delta' \Vdash\!\sim A \quad B \Vdash\!\sim \Gamma'}{\Delta[\Delta'] \Vdash\!\sim \Gamma[\Gamma'].}$$

Note that some of the traditional formulations of the Cut Rule may not hold, for example

Internal Cut

$$\frac{\Delta + A \mid\!\sim \Gamma \qquad \Delta \mid\!\sim A + \Gamma}{\Delta \mid\!\sim \Gamma}$$

Transitivity

$$\frac{\Delta \mid\!\sim A \qquad \Delta + A \mid\!\sim B}{\Delta \mid\!\sim B}$$

may not hold.

Further note that **Identity** can hold only for single formulas. I do not know whether the rule of identity formulated for single formulas A would imply the rule of identity for all common structures Δ.

Definition 6. Let $\mid\!\sim_1$ and $\mid\!\sim_2$ be two Tarski type S-consequence relations on structured databases with structures $\mathcal{M}_1$ and $\mathcal{M}_2$ respectively, based on the same language **L**, as indefinition 2. Assume that $\mid\!\sim_1$ and $\mid\!\sim_2$ are *compatible*, namely they satisfy for all A, B.

$A \mid\!\sim_1 B$ iff $B \mid\!\sim_2 A$

Define $\mid\!\sim_{1,2}$ to be the following Scott type S-consequence relation, called the *Minimal Amalgamation* of $\mid\!\sim_1$ and $\mid\!\sim_2$:

1. The data structures of $\vdash\!\sim_{1,2}$ are those of $\mathcal{M}_1$. The goal structures of $\vdash\!\sim_{1,2}$ are those of $\mathcal{M}_2$

2. Let $\Delta \vdash\!\sim_{1,2} \Gamma$ hold iff (definition) for some wff A we have that both $\Delta \vdash\!\sim_1 A$ and $\Gamma \vdash\!\sim_2 A$ hold.

Lemma 1. Let $\vdash\!\sim_1$ and $\vdash\!\sim_2$ be two compatible Tarski type consequence relations. Let $\Vdash\!\sim$ be their minimal amalgamation. Then $\Vdash\!\sim$ is a Scott type consequence relation.

Definition 7. Let **L** be a language and let $(\mathcal{M}_1, \mathcal{L}_1)$ and $(\mathcal{M}_2, \mathcal{L}_2)$ be structures for the data and goal respectively. Let $\vdash\!\sim_i$ i = 1, 2, be Tarski type S-consequence relations on the data structures $\mathcal{M}_i$ respectively with the goal structure being single wffs and let $\Vdash\!\sim$ be a Scott type S-consequence on both data structures $\mathcal{M}_1$ and goal structures $\mathcal{M}_2$.

(a) We say that $\Vdash\!\sim$ *agrees* with $\vdash\!\sim_1$ in the data (respectively $\Vdash\!\sim$ *agrees* with $\vdash\!\sim_2$ in the goal) iff for all Δ, Γ, A, (1) holds (respectively (2) holds).

(1) $\Delta \Vdash\!\sim A$ iff $\Delta \vdash\!\sim_1 A$

(2) $A \Vdash\!\sim \Gamma$ iff $\Gamma \vdash\!\sim_2 A$.

(b) We say that $\Vdash\!\sim$ agrees with $(\vdash\!\sim_1, \vdash\!\sim_2)$ iff $\Vdash\!\sim$ agrees with $\vdash\!\sim_1$ in the data and with $\vdash\!\sim_2$ in the goal.

Theorem 1. Let $\vdash\!\sim_1$ and $\vdash\!\sim_2$ be two compatible Tarski type S-consequence relations, i.e. satifying for all A, B, $A \vdash\!\sim_1 B$ iff $B \vdash\!\sim_2 A$. Then there exist two Scott-type consequence relations, a maximal $\Vdash\!\sim^+$ and a minimal $\Vdash\!\sim^-$ which agree with $(\vdash\!\sim_1, \vdash\!\sim_2)$. In other words, for any $\Vdash\!\sim$ which agrees with $(\vdash\!\sim_1, \vdash\!\sim_2)$ we have

1. for all Δ, Γ: $\Delta \Vdash\!\sim^- \Gamma$ implies $\Delta \Vdash\!\sim \Gamma$;
2. For all Δ, Γ: $\Delta \Vdash\!\sim \Gamma$ implies $\Delta \Vdash\!\sim^+ \Gamma$.

Example 6. (Symmetric Amalgamation) Given a Scott type S-consequence relation $\vdash\!\sim$, the two Tarski consequence relations derived from it, namely

$\Delta \vdash\!\sim_1 A$ iff $\Delta \vdash\!\sim A$
$\Gamma \vdash\!\sim_2 A$ iff $A \vdash\!\sim \Gamma$

may not be identical or isomorphic, even in the case where Δ and Γ are based on the same data structures. We say that the Scott type $\vdash\!\sim$ is a symmetrical amalgamation of the Tarski type $\vdash\!\sim_0$ iff the two derived consequence relations $\vdash\!\sim_1$ and $\vdash\!\sim_2$ are the same and equal to $\vdash\!\sim_0$. For a given $\vdash\!\sim_0$, it is always possible to construct the symmetrical amalgamation. We cannot take the naive identity $\vdash\!\sim_1 = \vdash\!\sim_2 = \vdash\!\sim_0$ because then the definition of amalgamation will yield

$\Delta \vdash\!\sim \Gamma$ iff for some A, $\Delta \vdash\!\sim_1 A$ and
$\Gamma \vdash\!\sim_2 A$ iff for some A, $\Delta \vdash\!\sim_0 A$ and $\Gamma \vdash\!\sim_0 A$.

If we let A = truth we get $\Delta \vdash\!\sim \Gamma$ for all Δ, Γ.
What is needed is an isomorphism function * such that

We obviously want to make use of both $\mid\!\sim_1$ and $\mid\!\sim_2$. The new consequence $\Vdash\!\sim$ should extend them both in some sense. Since both consequence relations are general, the only structures they share for certain are the unit formulae structures A. Given Δ, Γ and A, we know what $\Delta \mid\!\sim_1 A$ is and what $\Gamma \mid\!\sim_2 A$ is, but no more.

The next two definitions tell us what is the general notion of a Scott type S-consequence relation and how to form a Scott type $\Vdash\!\sim$ out of $\mid\!\sim_1$ and $\mid\!\sim_2$.

Definition 5. Let **L** be a language for wffs and let $(\mathcal{M}_1, \mathcal{L}_1)$ and $(\mathcal{M}_2, \mathcal{L}_2)$ be two languages for data and for goal structures respectively. Note that $\mathcal{L}_1$ and $\mathcal{L}_2$ come with their own respective notions of structural substitutions and structural additions $+_1$ and $+_2$ respectively.

A relation $\Delta \Vdash\!\sim \Gamma$ is said to be a (Scott type) S-consequence relation on the data and goal structures iff it satisfies the following two conditions:

Identity $A \Vdash\!\sim A$

Surgical Cut

$$\frac{\Delta[A] \Vdash\!\sim \Gamma[B] \quad \Delta' \Vdash\!\sim A \quad B \Vdash\!\sim \Gamma'}{\Delta[\Delta'] \Vdash\!\sim \Gamma[\Gamma'].}$$

Note that some of the traditional formulations of the Cut Rule may not hold, for example

Internal Cut

$$\frac{\Delta + A \mid\!\sim \Gamma \qquad \Delta \mid\!\sim A + \Gamma}{\Delta \mid\!\sim \Gamma}$$

Transitivity

$$\frac{\Delta \mid\!\sim A \qquad \Delta + A \mid\!\sim B}{\Delta \mid\!\sim B}$$

may not hold.

Further note that **Identity** can hold only for single formulas. I do not know whether the rule of identity formulated for single formulas A would imply the rule of identity for all common structures Δ.

Definition 6. Let $\mid\!\sim_1$ and $\mid\!\sim_2$ be two Tarski type S-consequence relations on structured databases with structures $\mathcal{M}_1$ and $\mathcal{M}_2$ respectively, based on the same language **L**, as indefinition 2. Assume that $\mid\!\sim_1$ and $\mid\!\sim_2$ are *compatible*, namely they satisfy for all A, B.

$A \mid\!\sim_1 B$ iff $B \mid\!\sim_2 A$

Define $\mid\!\sim_{1,2}$ to be the following Scott type S-consequence relation, called the *Minimal Amalgamation* of $\mid\!\sim_1$ and $\mid\!\sim_2$:

1. The data structures of $\mid\!\sim_{1,2}$ are those of $\mathcal{M}_1$. The goal structures of $\mid\!\sim_{1,2}$ are those of $\mathcal{M}_2$

2. Let $\Delta \mid\!\sim_{1,2} \Gamma$ hold iff (definition) for some wff A we have that both $\Delta \mid\!\sim_1$ A and $\Gamma \mid\!\sim_2$ A hold.

Lemma 1. Let $\mid\!\sim_1$ and $\mid\!\sim_2$ be two compatible Tarski type consequence relations. Let $\Vdash\!\sim$ be their minimal amalgamation. Then $\Vdash\!\sim$ is a Scott type consequence relation.

Definition 7. Let **L** be a language and let $(\mathcal{M}_1, \mathcal{L}_1)$ and $(\mathcal{M}_2, \mathcal{L}_2)$ be structures for the data and goal respectively. Let $\mid\!\sim_i$ i = 1, 2, be Tarski type S-consequence relations on the data structures $\mathcal{M}_i$ respectively with the goal structure being single wffs and let $\Vdash\!\sim$ be a Scott type S-consequence on both data structures $\mathcal{M}_1$ and goal structures $\mathcal{M}_2$.

(a) We say that $\Vdash\!\sim$ *agrees* with $\mid\!\sim_1$ in the data (respectively $\Vdash\!\sim$ *agrees* with $\mid\!\sim_2$ in the goal) iff for all Δ, Γ, A, (1) holds (respectively (2) holds).

(1) $\Delta \Vdash\!\sim$ A iff $\Delta \mid\!\sim_1$ A

(2) A $\Vdash\!\sim \Gamma$ iff $\Gamma \mid\!\sim_2$ A.

(b) We say that $\Vdash\!\sim$ agrees with $(\mid\!\sim_1, \mid\!\sim_2)$ iff $\Vdash\!\sim$ agrees with $\mid\!\sim_1$ in the data and with $\mid\!\sim_2$ in the goal.

Theorem 1. Let $\mid\!\sim_1$ and $\mid\!\sim_2$ be two compatible Tarski type S-consequence relations, i.e. satifying for all A, B, A $\mid\!\sim_1$ B iff B $\mid\!\sim_2$ A. Then there exist two Scott-type consequence relations, a maximal $\Vdash\!\sim^+$ and a minimal $\Vdash\!\sim^-$ which agree with $(\mid\!\sim_1, \mid\!\sim_2)$. In other words, for any $\Vdash\!\sim$ which agrees with $(\mid\!\sim_1, \mid\!\sim_2)$ we have

1. for all Δ, Γ: $\Delta \Vdash\!\sim^- \Gamma$ implies $\Delta \Vdash\!\sim \Gamma$;
2. For all Δ, Γ: $\Delta \Vdash\!\sim \Gamma$ implies $\Delta \Vdash\!\sim^+ \Gamma$.

Example 6. (Symmetric Amalgamation) Given a Scott type S-consequence relation $\mid\!\sim$, the two Tarski consequence relations derived from it, namely

$\Delta \mid\!\sim_1$ A iff $\Delta \mid\!\sim$ A
$\Gamma \mid\!\sim_2$ A iff A $\mid\!\sim \Gamma$

may not be identical or isomorphic, even in the case where Δ and Γ are based on the same data structures. We say that the Scott type $\mid\!\sim$ is a symmetrical amalgamation of the Tarski type $\mid\!\sim_0$ iff the two derived consequence relations $\mid\!\sim_1$ and $\mid\!\sim_2$ are the same and equal to $\mid\!\sim_0$. For a given $\mid\!\sim_0$, it is always possible to construct the symmetrical amalgamation. We cannot take the naive identity $\mid\!\sim_1 = \mid\!\sim_2 = \mid\!\sim_0$ because then the definition of amalgamation will yield

$\Delta \mid\!\sim \Gamma$ iff for some A, $\Delta \mid\!\sim_1$ A and
$\Gamma \mid\!\sim_2$ A iff for some A, $\Delta \mid\!\sim_0$ A and $\Gamma \mid\!\sim_0$ A.

If we let A = truth we get $\Delta \mid\!\sim \Gamma$ for all Δ, Γ.
What is needed is an isomorphism function * such that

$A^{**} = A$ and $\Gamma \mid\!\sim_2 A$ iff $\Gamma^* \mid\!\sim_1 A^*$.

We thus get

$\Delta \mid\!\sim \Gamma$ iff for some A, $\Delta \mid\!\sim_1 A$ and $\Gamma^* \mid\!\sim_1 A^*$

now since we also have

$A \mid\!\sim_1 B$ iff $B \mid\!\sim_2 A$

we get that * must satisfy

$A \mid\!\sim_1 B$ iff $B^* \mid\!\sim_1 A^*$.

In case of classical logic the mapping * is negation

$A \vdash B$ iff $\neg B \vdash \neg A$.

The above examples have shown how to extend S-consequence relations to ones with a more complex goal structure by minimal amalgamation. When the original consequence relation $\mid\!\sim$ satisfies the Deduction Theorem, it is possible to extend $\mid\!\sim$ by making use of this fact. The Deduction Theorem is similar to a left to right shift operator (studied in a later section). Thus by using shift operators we can give meaning to $\Delta \mid\!\sim \Gamma$ for Γ more complex, in terms of shifting Γ to the left and reducing the problem to a known one.

Thus for example one may write $\Delta \mid\!\sim \Gamma$ iff some boolean combination $\Delta_i \mid\!\sim \Gamma_i$ ... holds, where each Γ_i has less elements than Γ.

Classical logic can be viewed to be a special case of reducing $\{A_i\} \vdash_C \{B_j, C\}$ to $\{A_i, \neg C\} \vdash_C \{B_j\}$.

The Intuitionistic consequence, arising from the Kripke Semantics, cannot be viewed in this manner. We have no means of shifting wffs from the right hand side to the left hand side.

Let $\mid\!\sim'$ be a consequence relation with $\mathcal{M}'_1$ and $\mathcal{M}'_2$ being the structures for the data and goals respectively. Let $\mid\!\sim''$ be another consequence relation with structures $\mathcal{M}''_1$ and $\mathcal{M}''_2$ respectively. We want to define the composition $\mid\!\sim^*$ of $\mid\!\sim'$ and $\mid\!\sim''$. The data structures (goal structures) of $\mathcal{M}^*_1$ ($\mathcal{M}^*_2$) are obtained from $\mathcal{M}'_1$ and $\mathcal{M}''_1$ (respectively $\mathcal{M}'_2$ and $\mathcal{M}''_2$) by the repeated substitution of one data structure inside the other.

If the structures in $\mathcal{M}'_1$ are lists of points $(x_1,\dots,x_n)$ and the structures in $\mathcal{M}''_1$ are sets then the structures in $\mathcal{M}^*_1$ are hereditary lists of non-empty sets of points $(X_1,\dots,X_n)$. If $\mathcal{M}'_1$ contains multisets of points then $\mathcal{M}^*_1$ is a hereditary set of multisets of non-empty sets of points.

Definition 8. (Composition of two S-consequence relations) Let $\mid\!\sim'$ and $\mid\!\sim''$ be two consequence relations with structures $(\mathcal{M}'_1, \mathcal{M}'_2)$ and $(\mathcal{M}''_1, \mathcal{M}''_2)$ respectively. We assume further that we have available a notion of substitution of a τ' structure inside a τ'' structure and vice versae as needed below. We define the composition $\mid\!\sim^*$ of $\mid\!\sim'$ and $\mid\!\sim''$ as follows:

1. The structures of $\mathcal{M}^*_i$ of $\vdash\!\!\sim^*$, i = 1, 2 are defined by induction.

(a) Let $\mathcal{M}^*_i(0) = \mathcal{M}'_i \cup \mathcal{M}''_i$.

(b) Assume $\mathcal{M}^*_i(k)$, for k < n are all defined. Let $\tau \in \mathcal{M}^*_i(k)$ for some k < n and $\tau'(x)$ (respectively $\tau''(x)$) is in $\mathcal{M}'_i$ (respectively $\mathcal{M}''_i$). Then $\tau'(\tau)$ (resp $\tau''(\tau)$) is in $\mathcal{M}^*_i(n)$.

(c) Let $\mathcal{M}^*_i = \bigcup \mathcal{M}^*_i(n)$.

Note: It is easy to show that if $\tau_1(x) \in \mathcal{M}^*_i$ and $\tau_2 \in \mathcal{M}^*_i$ then $\tau_1(\tau_2) \in \mathcal{M}^*_i$.

2. Let $\Delta = (\tau_1, \delta_1)$ and $\Gamma = (\tau_2, \delta_2)$ be two databases for $\vdash\!\!\sim^*$.

We can assume by definition that for some $\alpha_1, \beta_1, \alpha_2, \beta_2, \varepsilon, \eta$:

$\tau_1 = \beta_1(\alpha_1)$ hence $\Delta = \Delta_0(\varepsilon)$

$\tau_2 = \beta_2(\alpha_2)$ hence $\Gamma = \Gamma_0(\eta)$, where

$\Delta_0(x) = (\beta_1(x), \delta_1)$, $\Gamma_0(x) = (\beta_2(x), \delta_2)$

$\varepsilon = (\alpha_1, \delta_1)$, $\eta = (\alpha_2, \delta_2)$

where β_1, β_2 are either in $\mathcal{M}'_1$ or in $\mathcal{M}''_i$ for i = 1, 2 respectively.

Note that δ_i do not give a value to x.

We say that $\Delta \vdash\!\!\sim^* \Gamma$ iff for some C, D

$\varepsilon = (\alpha_1, \delta_1) \vdash\!\!\sim^* C$

$\eta = (\alpha_2, \delta_2) \vdash\!\!\sim^* D$ and

$\Delta_0 [C] \vdash\!\!\sim^* \Gamma_0[D]$

Let us summarise our current view:

- A logical -language **L** is defined by introducing the notion of well formed formulas and by further stating a family of data structures and defining the notions of substitution on the data structure, and of adding (combining) data structures.

- Minimal conditions for a relation between data structures and formulas to be a consequence relation are those of **Identity** and **Surgical Cut**. The smallest consequence relation (if it exists) for a language **L** is denoted by **S** (**L**).

- The smallest structured consequence relation for the langauge with $\rightarrow$ only, can be extended either in the direction of non-monotonic consequence relations or in the direction of Substructural Logics or both.

- We need and can develop adequate semantics for **S** (**L**) in the spirit of [3], see [34].

3. Structural Connectives

Let $\vdash\!\!\sim$ be a Scott type consequence relation on a language **L**. The consequence relation involves structures for the data and structures for goal. The purpose of this section is to show that given these structures, there are some natural connectives, called structural connectives, which one may wish to add into the language **L**, with porperties dictated by the structure. In fact, the entire consequence relation may be in some cases (e.g. linear logic, see next section) no more and no less than a reflection (via the structural connectives) of the properties of the data structures involved.

Let $\vdash\!\sim$ be any Scott type consequence relation. Let $\mathcal{M}_1$ be structures for the data and let $\mathcal{M}_2$ be the structures for the goals. Let $+_1$ be structural addition for the data and $+_2$ for the goal. We can thus write

$$\Delta +_1 A \vdash\!\sim B +_2 \Gamma$$

which may or may not hold.

We assume nothing special about $\vdash\!\sim$, only that it satisfies the mimimal properties of **Identity** and **Surgical Cut**. We do not assume any special connectives in the language **L** of $\vdash\!\sim$, and we aim to introduce several structural connectives.

It might be useful to think in terms of two examples, namely concatenation logic, where the structures are lists, and linear logic, where the structures are multisets. Both cases would have $\rightarrow$ in the language **L**. To be even more specific, think of the Tarski type consequence relations to be the relations $\Vdash_{CL}$ and $\Vdash_{LL}$ for **CL** and for **LL** of example 3 respectively, and assume that $\vdash\!\sim$ is some Scott extension of one of them which agrees with it, the goal structures being the same as the data structures. Note further that there may be more than one such $\vdash\!\sim$ agreeing with the Tarski type $\Vdash_{CL}$ or $\Vdash_{LL}$ of example 3, and that in the sequel we may choose one such $\vdash\!\sim$ with special symmetry conditions.

Let us begin with a general Scott type consequence $\vdash\!\sim$.

The first thing to bear in mind is that in the general case, the structural operators "$+_1$" and "$+_2$" which come with $\vdash\!\sim$ are in the metalevel, on the structures of $\mathcal{M}_1$ and $\mathcal{M}_2$ respectively, and not in the language **L** of $\vdash\!\sim$ itself. For the special case of **CL**, "$+_i$" is concatenation and for the case of **LL**,"$+_i$" is multiset union. Let us now add another metalevel operator, a sort of notational shift operator "o", having the following properties:

$$\Delta +_1 A^o \vdash\!\sim \Gamma \text{ means the same as } \Delta \vdash\!\sim A +_2 \Gamma$$

and

$$\Delta \vdash\!\sim B^o +_2 \Gamma \text{ means the same as } \Delta +_1 B \vdash\!\sim \Gamma$$

In fact, $\Delta +_1 A^o \vdash\!\sim B^o +_2 \Gamma$ should be the same as $\Delta +_1 B \vdash\!\sim A +_2 \Gamma$.

Thus the operator "o" in "A^o" is used to indicate, in the metalevel, where the formula A should be. It has the status of a metalevel annotation, indicating that although the formula A appears in one place in the structure, it should be in another place. Thus $\Delta +_1 A^o \vdash\!\sim \Gamma$ really says through the annotation "o" on A, that A is really on the right hand side, namely $\Delta \vdash A +_2 \Gamma$. For general structures the annotation may be problematic. If $\Delta[x]$ indicates that x stands somewhere in the structure Δ, then we know what $\Delta[A]$ means. We also have to say what $\Delta[A^o]$ means. We have to specify the following:

1. Since A^o in $\Delta[A^o]$ means that although A^o shows up in $\Delta[x]$ at the place x (i.e. we have $\Delta[A^o/x]$), A should really be somewhere else, we need to specify where A should be.

2. If A^o is not at place x in $\Delta[x]$, then there is *nothing* at x. Hence our notion of substitution should also allow for empty substitution $\Delta[\emptyset/x]$, which is really deletion.

If both Δ and Γ are multisets, as in the case of linear logic, then we have no such problem, otherwise a precise definition is required. Even in the case of **CL**, this is not simple, and requires care.

In **CL**, the data structures are lists $(A_1,\ldots,A_n)$. Of course $(A_1,\ldots,A_n, B)$ is generally not the same data structure as $(B, A_1,\ldots,A_n)$. We can mark B as Bo to mean that B should be *shifted* to the other end of the list. Thus

$(A_1,\ldots,A_n, B^o) =_{def} (B, A_1,\ldots,A_n)$
$(B^o, A_1,\ldots,A_n) =_{def} (A_1,\ldots,A_n, B)$.

Note that $\Delta_1 +_2 \Delta_2$ (and similarly $\Gamma_1 +_1 \Gamma_2$) is not meaningful, since $+_2$ (resp $+_1$) is a goal (data) structural addition and cannot be applied to the data (goal).

So far we have introduced several meta operations. We now want to put them in the object level. We have seen in connection with the Deduction Theorem in the previous section that it is possible to take a meta operation and add a connective for it in the object language. We now do just that for "$+_1$", "$+_2$" and "o" of the metalevel. We denote the object level connectives by "$\oplus_1$", "$\oplus_2$" and "$\odot$", respectively.

Our purpose is to study their obvious properties. The connection between {A + B} and {A ⊕ B} is that they must be declaratively identical. This means that A ⊕ B should behave essentially like A + B in all $\mid\!\sim$ contexts. We thus have the following rules:

Identity

$A +_1 B \mid\!\sim A \oplus_1 B$
$A \oplus_2 B \mid\!\sim A +_2 B$

The other direction of the identity of $A \oplus_i B$ with $A +_i B$, cannot be written directly but can be characterised by a Cut rule:

Surgical Cut

$$\frac{\Delta_1 +_1 (A +_1 B) +_1 \Delta_2 \mid\!\sim \Gamma_1 +_2 (C +_2 D) +_2 \Gamma_2}{\Delta_1 +_1 (A \oplus_1 B) +_1 \Delta_2 \mid\!\sim \Gamma_1 +_2 (C \oplus_2 D) +_2 \Gamma_2}$$

We believe that the object level counterparts of the structural connectives of any system can always be conservatively added to the system. We may have to assume some mild and very reasonable sufficient conditions.

Example 7. In the case of **CL**, if we add the structural connective ⊕ to the language, we get the following system **CL** (⊕), with the following two rules:

$(A, B) \mid\!\sim A \oplus B$

which we get from **Identity**, and we should also have the following from the **Cut Rule**

$$\frac{(A_1,\dots,A_n,\ A,\ B,\ B_1,\dots,B_m) \vdash C}{(A_1,\dots,A_n,\ A \oplus B,\ B_1,\dots,B_m) \vdash C}$$

These we call **Structural Rules** for the structural connective $\oplus$, because they arise from the fact that $\oplus$ is no more than an object level reflection of the structured database operation +.

In classical logic the data structures are sets and conjunction "$\wedge$" serves as the structural connective. The goal structures are also sets and "$\vee$" serves as the structural connectives.

In case of **CL** we also have a **Right hand side Deduction Theorem**, which, when combined with the structural connective $\oplus$ will yield rules like

$$A \oplus B \vdash C \text{ iff } A \vdash B \rightarrow C$$

and

$$\emptyset \vdash A \rightarrow (B \rightarrow A \oplus B).$$

We now have to check which rules for $\odot$ follow from the properties of shift. We know that $\Delta +_1 A^{\odot} \vdash B^{\odot} +_2 \Gamma$ should hold iff $\Delta +_1 B \vdash A +_2 \Gamma$. We immediately get

$$A^{\odot\,\odot} \vdash A$$
$$A \vdash A^{\odot\,\odot}.$$

Consider

$$\Delta +_1 C^{o} +_1 D^{o} \vdash \Gamma$$

which is the same as

$$\Delta \vdash (C \oplus_2 D) +_2 \Gamma$$

which is the same as

$$\Delta +_1 (C \oplus_2 D)^{o} \vdash \Gamma$$

We thus have that $(C \oplus_2 D)^{o}$ must be the same as $C^{o} \oplus_1 D^{o}$. Similarly $(C \oplus_1 D)^{o}$ must behave the same as $C^{o} \oplus_2 D^{o}$. We thus get

$$(C \oplus_2 D)^{\odot} \vdash C^{\odot} \oplus_1 D^{\odot}$$
$$(C \oplus_1 D)^{\odot} \vdash C^{\odot} \oplus_2 D^{\odot}$$

We still have a problem of what to do (what meaning we can prove or have) to $A \oplus_1 B$ when it appears on the right hand side and $C \oplus_2 D$ when it appears on the left hand side. The typical expression is:

$$C \oplus_2 D \vdash A \oplus_1 B$$

clearly it should be the same as

$$(A \oplus_1 B)^{\odot} \mid\!\sim (C \oplus_2 D)^{\odot}$$

which is the same as

$$A^{\odot} \oplus_2 B^{\odot} \mid\!\sim C^{\odot} \oplus_1 D^{\odot}$$

which does not help, as again we get the wrong "$\oplus$" on the wrong side.

In general, there is nothing to be done. However, if we have the object level "$\rightarrow_r$" with the Deduction Theorem holding, a few more equivalences can be derived.

Consider

$$A +_1 B \mid\!\sim C$$

this holds iff

$$A \mid\!\sim B \rightarrow_r C$$

but also iff

$$A \mid\!\sim B^{o} +_2 C.$$

Hence

$B \rightarrow_r C$ is equivalent to $B^{o} \oplus_2 C$.

The difference between the shift and the Deduction theorem is that the connective "$\rightarrow_r$" is in the object language **L**, while our "$+_2$" and "o" are in the metalevel. $\Delta \mid\!\sim B^{o} +_2 C$ is just another way of writing, in the metalevel, $\Delta +_1 B \mid\!\sim C$.

However, if we do have the object level connectives,"$\oplus_1$", and "$\oplus_2$", we can write down equivalences. We therefore get that

$A \oplus_2 B$ is equivalent to $A^{\odot} \rightarrow_r B$
$A \oplus_1 B$ is equivalent to $(A \rightarrow_r B^{\odot})^{\odot}$

Thus "$\oplus_1$", "$\oplus_2$" are reducible, in a system containing "$\rightarrow_r$", to the connectives "$\rightarrow_r$" and "$\odot$"

Let us now pause and summarise what we have learnt so far.

Suppose we start with a consequence relation $\mid\!\sim$ with $+_1$ for data and $+_2$ for goals. Suppose we introduce a shift operation which shifts from the right hand side of the data to the left hand side of the goals and vice versae, and suppose that we have an object level connectives $\odot$ for the shift and $\rightarrow_r$ satisfying right hand side Deduction Theorem. Then we can define in the object level the two operators "$\oplus_1$" and "$\oplus_2$" using "$\rightarrow_r$" and the object level "$\odot$" as follows:

$A \oplus_2 B = \text{def } A^{\odot} \rightarrow_r B$
$A \oplus_1 B = \text{def } (A \rightarrow_r B^{\odot})^{\odot}.$

The above discussion showed how to add a structural connective ⊕ to reflect the notion +, which is part of the defintion of any consequence relation. Any specific consequence relation allows for a family $\mathcal{M}$ of structures. Any of these structures $\tau \in \mathcal{M}$ can be reflected in the object level by a special connective $\boxed{\tau}$. The way it is done is described in the next definition.

Definition 9. (Left Structural Connectives and Left Structural Rules) Let $\mathcal{M}$ be a class of structures and let $\mid\!\sim$ be a consequence relation structured on $\mathcal{M}$, as defined in 2. Let τ be a fixed structure in $\mathcal{M}$ and assume τ has n elements $a_1,...,a_n$. Indicate this fact by writing $\tau(a_1,...,a_n)$. Enrich the language of $\mid\!\sim$ with an additional n-place connective $\boxed{\tau}\ (A_1,...,A_n)$ with the following *Left Structural Rules for* $\boxed{\tau}$.

$\boxed{\tau}$ *Identity*

$(\tau(a_1,..., a_n), \delta) \mid\!\sim \boxed{\tau}\ (\delta(a_1),...,\delta(a_n))$.

$\boxed{\tau}$ *Surgical Cut*

Let $\Delta_1 = (\tau_1, \delta_1)$ and $\Delta = (\tau, \delta)$. Let $x \in \tau_1$ and let $\delta_1(x) = B$. Let $\Delta_2 = (\tau_2, \delta_2)$ be the result of structural substitution of Δ in Δ_1, i.e. $\Delta_2 = \Delta_1[B/\Delta]$.

This means that $\tau_2 = \lambda x \tau_1(x)[\tau]$.

Consider $\Delta'_1 = (\tau_1(x), \delta'_1)$ which is just like Δ_1 except that $\delta'_1(x) = \boxed{\tau}\ (\delta(a_1),...,\delta(a_n))$. Thus $\Delta'_1 = \Delta_1[B/\boxed{\tau}\ (\delta(a_1),...,\delta(a_n))]$.

The **Surgical Cut Rule** is therefore

$$\frac{\Delta_1[B/\Delta] \mid\!\sim A}{\Delta_1[B/\boxed{\tau}\ (\delta(a_1),...,\delta(a_n))] \mid\!\sim A}$$

or if formulated with a slight abuse of notation:

$$\frac{\lambda x \tau_1(x)[\tau] \mid\!\sim A}{\lambda x \tau_1(x)[\boxed{\tau}\ (\delta(a_1),...,\delta(a_n))] \mid\!\sim A}$$

The cut rule really ensures the other direction of the identity, by forcing $\boxed{\tau}$ to behave properly.

We are now ready to explain what a *Structural Shift Connective* is and what **Structural Shift Rules** are.

Definition 10. (Structural Shift Connectives) Let $\mid\!\sim$ be a Scott type S-consequence relation and let $\mathcal{M}_i$ and $\mathcal{L}_i$ be as in definition 5. Let ϕ_i be the wff of $\mathcal{L}_i$ each identifying a unique point in each $\tau_i \in \mathcal{M}_i$. Further assume that for each τ

$(a_1,\ldots,a_n)$ there exist a unique τ_2 $(a_1,\ldots,a_n, z)$ such that $\tau = \tau_2 \parallel \phi_2$. Assume $\tau' = \tau_1 \parallel \phi_1$. We can write $\tau' = f_1(\tau_1)$, and $\tau_2 = f_2(\tau)$ where f_i are function symbols for the functional dependence which exists.

Let $\odot^{\phi_1}_{\phi_2}$ be a new unary connective to the language of $\mid\!\sim$. Enrich $\mid\!\sim$ with this connective and the following structural shift rule:

(ϕ_1, ϕ_2) **Structural Shift Rule**

Let $\Delta_1 = (\tau_1(x), \delta_1)$ be the data structure with x the point identified in τ_1 by ϕ_1, and let $\Delta_2 = (\tau_2(y), \delta_2)$ be the goal structure with y the point identified by ϕ_2. Assume that $\delta_1(x)$ and $\delta_2(y)$ satisfy that either $\delta_1(x) = \odot^{\phi_1}_{\phi_2}$ A and $\delta_2(y) = A$ or $\delta_1(x) = A$ and $\delta_2(y) = \odot^{\phi_1}_{\phi_2}$ A. For each of the above the following holds:

$$\Delta_1 \mid\!\sim \Delta_2 \parallel \phi_2 \text{ iff } \Delta_1 \parallel \phi_1 \mid\!\sim \Delta_2$$

The functions induced by ϕ_1, ϕ_2 define the two places, and the shift rule tells us that $\odot^{\phi_1}_{\phi_2}$ A shifts A from one place to the other place.

It is worth noting that with a bit of abuse of notation, the shift operation can be written as

1. $\Delta_1[\odot\ A] \mid\!\sim \Delta_2[\emptyset]$ iff $\Delta_1[\emptyset] \mid\!\sim \Delta_2[A]$
2. $\Delta_1[A] \mid\!\sim \Delta_2[\emptyset]$ iff $\Delta_1[\emptyset] \mid\!\sim \Delta_2[\odot\ A]$

where $\emptyset$ is the empty set and its substitution at point x in $\Delta[x]$ means deletion.

Example 8. Consider **CL** with its list structures. Let ϕ_1 define the last element of a list and let ϕ_2 define the second element of a list. Then if $\odot$ denotes $\odot^{\phi_1}_{\phi_2}$ we get to the following two shift rules:

1. $(A_1,\ldots,A_m, \odot B) \mid\!\sim (C_1,\ldots,C_n)$
   iff $(A_1,\ldots,A_m) \mid\!\sim (C_1, B, C_2,\ldots,C_n)$
2. $(A_1,\ldots,A_n) \mid\!\sim (C_1, \odot B, C_2,\ldots,C_n)$
   iff $(A_1,\ldots,A_n, B) \mid\!\sim (C_1,\ldots,C_n)$

The shift operation suggest a new structural connective $\boxed{!}$ A. Its meaning can be motivated by the following observation. Given $\Delta_1(x)$ and $\Delta_2(y)$ the shift rule, stated intuitively, says for example that:

$$\Delta_1[A] \mid\!\sim \Delta_2(\emptyset) \text{ iff } \Delta_1[\emptyset] \mid\!\sim \Delta_2[\odot\ A].$$

We understand the operation as shifting A from Δ_1 into Δ_2. After the shift, A is no longer present in Δ_1 but is recorded in Δ_2. That is why we write $\Delta_1[\emptyset]$. If we do not take A out Δ_1, we get a different rule namely

$$\Delta_1[\boxed{!}\ A] \mid\!\sim \Delta_2[\emptyset] \text{ iff } \Delta_1[\boxed{!}\ A] \mid\!\sim \Delta_2[\odot\ A].$$

This means that although A is supposed to "shift" to the right hand side, it stays while "duplicating" itself. The connective "$\boxed{!}$" indicates the duplications property.

Example 9. Consider the previous example where we had

$$(A_1,\dots,A_m, B) \mathrel{|\!\sim} (C_1,\dots,C_n)$$
$$\text{iff } (A_1,\dots,A_n) \mathrel{|\!\sim} (C_1{}^{\odot}, B, C_2,\dots,C_n)$$

If the formula B does not delete itslef when shifting, but duplicates itself we indicate this by writing $\boxed{!}$ B instead of B.

$$(A_1,\dots,A_n \boxed{!}\ B) \mathrel{|\!\sim} (C_1,\dots,C_n)$$
$$\text{iff } (A_1,\dots,A_m, \boxed{!}\ B) \mathrel{|\!\sim} (C_1 \odot B, C_2,\dots,C_n)$$

Thus $\boxed{!}$ B at the last place on the left simply indicates an infinite number of B's. Of course we can shift the entire $\boxed{!}$ B to the right and get

$$(A_1,\dots,A_m, \boxed{!}\ B) \mathrel{|\!\sim} (C_1,\dots,C_n)$$
$$\text{iff } (A_1,\dots,A_m) \mathrel{|\!\sim} (C_1 \odot \boxed{!}\ B, C_2,\dots,C_n)$$

Some properties of $\boxed{!}$ and $\odot$ can be obtained from the above meaning, especially if other connectives are present, for example

$$\boxed{!}\ B + B \equiv \boxed{!}\ B$$

Another structural connective is the proof-net like connective, which identifies copies. Consider again $\Delta_1[x]$ and $\Delta_2[y]$. Using a shift connective, we can either put $x = A$ and $y = \varnothing$ or put $x = \varnothing$ and $y = \odot A$. In either case we get A on the left ($\Delta_1[A]$) and nothing on the right.

Another possibility is to put A on the right and nothing on the left, i.e. $\Delta_1[\varnothing]$, $\Delta_2[A]$. We can write this possibility also as $\Delta_1[\odot A]$, $\Delta_2[\varnothing]$.

What we should *not* write is

$$\Delta_1[\odot A] \mathrel{|\!\sim} \Delta_2[A]$$

because here we are duplicating A twice on the right. We could of course add an additional marker say † †, to indicate that these two copies of A should really be *one* copy. So we can write:

$$\Delta_1[\odot \dagger_A\dagger] \mathrel{|\!\sim} \Delta_2[\dagger_A\dagger]$$

where † † marks the fact that there is only one copy. Thus($\odot \dagger_A\dagger$, $\dagger_A\dagger$) is not the same as ($\dagger_A\dagger$, $\odot \dagger_A\dagger$) because the first corresponds to A on the right and the second corresponds to A on the left.

Example 10. To continue the previous example,

$$(A_1,\ldots,A_n, \odot\ \dagger_A\dagger) \mathrel{|\!\sim} (C_1, \dagger_A\dagger, C_2,\ldots,C_n)$$

is the same as

$$(A_1,\ldots,A_m) \mathrel{|\!\sim} (C_1, A, C_2,\ldots,C_n)$$

while

$$(A_1,\ldots,A_m, \dagger_A\dagger) \mathrel{|\!\sim} (C_1, \odot\ \dagger_A\dagger, C_2,\ldots,C_m)$$

is the same as

$$(A_1,\ldots,A_m, A) \mathrel{|\!\sim} (C_1,\ldots,C_n)^3.$$

We now turn out attention to the so called "additive" connectives. As before, we start with a general consequence relation $\mathrel{|\!\sim}$.

Consider now the case where we have that both $\Delta \mathrel{|\!\sim} \Gamma[A_1]$ and $\Delta \mathrel{|\!\sim} \Gamma[A_2]$ hold. We can abbreviate the above situation by writing in the metalevel $\Delta \mathrel{|\!\sim} \lambda x\Gamma(x)[A_1 \wedge A_2]$ or just $\Delta \mathrel{|\!\sim} \Gamma[A_1 \wedge A_2]$ where "$\wedge$" is a metaleval symbol, just like "$+_1$", "$+_2$" and "o". Similarly if we have $\Delta[A_1] \mathrel{|\!\sim} \Gamma$ and $\Delta[A_2] \mathrel{|\!\sim} \Gamma$ we can abbreviate it using "$\vee$" and write $\Delta[A_1 \vee A_2] \mathrel{|\!\sim} \Gamma$. Note that "$\wedge$" and "$\vee$" only abbreviate other notation and are not operations on the data. We could modify the data structures if we want to accommodate the respective operations but that is not necessary.

Note that $\Delta[A_1 \wedge A_2] \mathrel{|\!\sim} \Gamma$ is not meaningful, as we have not said what it means when "$\wedge$" appears on the left. Similarly for $\Delta \mathrel{|\!\sim} \Gamma\, [A_1 \vee A_2]$.

The following hold by definition

$\Delta \mathrel{|\!\sim} A \wedge B$ iff $\Delta \mathrel{|\!\sim} A$ and $\Delta \mathrel{|\!\sim} B$
$A \vee B \mathrel{|\!\sim} \Gamma$ iff $A \mathrel{|\!\sim} \Gamma$ and $B \mathrel{|\!\sim} \Gamma$.

Example 11. Suppose we add the structural connectives $\boxed{\vee}$ and $\boxed{\wedge}$ to the language; what are the obvious properties it must satisfy? In the general case we cannot say much beyond the obvious, so let us see what we get in the special case of linear logic. Here the Deduction Theorem comes into play:

1. $\Delta \mathrel{|\!\sim} A \rightarrow B \boxed{\wedge} C$ iff $\Delta, A \mathrel{|\!\sim} B \boxed{\wedge} C$
   iff $\Delta, A \mathrel{|\!\sim} B$ and $\Delta, A \mathrel{|\!\sim} C$
   iff $\Delta \mathrel{|\!\sim} A \rightarrow B$ and $\Delta \mathrel{|\!\sim} A \rightarrow C$
   iff $\Delta \mathrel{|\!\sim} (A \rightarrow B) \boxed{\wedge} (A \rightarrow C)$.
2. $\Delta \mathrel{|\!\sim} A \boxed{\vee} B \rightarrow C$
   iff $\Delta, A \boxed{\vee} B \mathrel{|\!\sim} C$
   iff $\Delta, A \mathrel{|\!\sim} C$ and $\Delta, B \mathrel{|\!\sim} C$
   iff $\Delta \mathrel{|\!\sim} A \rightarrow C$ and $\Delta \mathrel{|\!\sim} B \rightarrow C$

iff $\Delta \mid\!\sim (A \to C) \boxed{\wedge} (B \to C)$.

3. From the Identity Rule we get, since $A \boxed{\vee} B \mid\!\sim A \boxed{\vee} B$ that

$A \mid\!\sim A \boxed{\vee} B$

$B \mid\!\sim A \boxed{\vee} B$

4. Similarly since by the Identity Rule $A \boxed{\wedge} B \mid\!\sim A \boxed{\wedge} B$ we get that:

$A \boxed{\wedge} B \mid\!\sim A$

$A \boxed{\wedge} B \mid\!\sim B$

5. Since $A \mid\!\sim A \boxed{\vee} B$ we get $(A \boxed{\vee} B)^{\odot} \mid\!\sim A^{\odot}$ and similarly $(A \boxed{\vee} B)^{\odot} \mid\!\sim B^{\odot}$. Hence we get $(A \boxed{\vee} B)^{\odot} \mid\!\sim A^{\odot} \boxed{\wedge} B^{\odot}$.

6. Since $A \boxed{\wedge} B \mid\!\sim A$ we get $A^{\odot} \mid\!\sim (A \boxed{\wedge} B)^{\odot}$. Similarly $B^{\odot} \mid\!\sim (A \boxed{\wedge} B)^{\odot}$ and therefore $A^{\odot} \boxed{\vee} B^{\odot} \mid\!\sim (A \boxed{\wedge} B)^{\odot}$.

7. Since $(A \to C) \boxed{\wedge} (B \to C) \mid\!\sim A \to C$ and $A \oplus_1 (A \to C) \mid\!\sim C$ we get by **Surgical Cut** that $A \oplus_1 ((A \to C) \boxed{\wedge} (B \to C)) \mid\!\sim C$ and similarly $B \oplus_1 ((A \to C) \boxed{\wedge} (B \to C)) \mid\!\sim C$ hence $(A \boxed{\vee} B) \oplus_1 ((A \to C) \boxed{\wedge} (B \to C)) \mid\!\sim C$ and therefore $(A \to C) \boxed{\wedge} (B \to C) \mid\!\sim A \boxed{\vee} B \to C$.

So far our study of structural connectives and shift connectives used no special properties of $\mid\!\sim$ beyond the existence of $\to_r$ in the object language. We shall see that we are going to need some symmetry assumptions on $\mid\!\sim$. Suppose we start with the Tarski $\mid\!\sim_{LL}$ of example 3. and assume that $\mid\!\sim_1$ and $\mid\!\sim_2$ are both Scott type and are compatible with $\mid\!\sim_{LL}$. Everything we have said and done so far would apply to both $\mid\!\sim_1$ and $\mid\!\sim_2$ equally. We can thus add the structural connective "$\odot$" and get all the equivalences mentioned earlier to hold. However there might be a difference between the case of $\mid\!\sim_1$ and that of $\mid\!\sim_2$, depending on their properties.

Example 12. Consider the consequence relation $\mid\!\sim_{LL^1}$ defined by

$\Delta \mid\!\sim_{LL^1} A$ iff for some $B \in \Delta$, $A \mid\!\sim_{LL} B$,

where Δ is a multiset. This is a consequence relation.

Let $\mid\!\sim_1$ be the minimal amalgamation of $\mid\!\sim_{LL}$ and $\mid\!\sim_{LL^1}$. Clearly $\Delta \mid\!\sim_1 \Gamma$ iff for some $A \in \Gamma$, $\Delta \mid\!\sim_{LL} A$ holds. This means that $\mid\!\sim_1$ is monotonic in Γ.

Let us add to the $\mid\!\sim_1$ the shift "$\odot$" and consider the following: $A \mid\!\sim_1 B, C$ iff $A \mid\!\sim_1 B^{\odot} \to C$ iff $A, B^{\odot} \mid\!\sim_1 C$. If we continue the chain using the fact that $\mid\!\sim_1$ agrees with $\mid\!\sim_{LL}$ we get that the above holds iff $A, B^{\odot} \mid\!\sim_{LL} C$. We can now get that $\mid\!\sim_{LL}$ is monotonic in $B^{\odot}$, i.e. $\Delta \mid\!\sim_{LL} C$ implies $\Delta, B^{\odot} \mid\!\sim_{LL} C$. This is so because $\Delta \mid\!\sim_{LL} C$ implies $\Delta \mid\!\sim_1 B, C$ and hence $\Delta, B^{\odot} \mid\!\sim_{LL} C$ from the equivalence chain. Although $\mid\!\sim_{LL}$

itself is not monotonic, there is nothing wrong with monotonicity in $B^{\odot}$, because we know that Δ, $B^{\odot} \mid\!\sim_{LL}$ C really means $\Delta \mid\!\sim_1$ B, C. However, it does force us syntactically to note which wff is of the form B and which is of the form $B^{\odot}$, and excludes writing $B^{\odot\odot}$ instead of B. Clearly this is not desirable. We therefore need to assume further properties on the $\mid\!\sim$ which agrees with $\mid\!\sim_{LL}$, namely the property that it is *symmetrical*, namely $A \mid\!\sim \Gamma$ iff $\Gamma^{\odot} \mid\!\sim_{LL} A^{\odot}$.

In terms of definition 6, we want $\mid\!\sim$ to be the amalgamation of $\mid\!\sim_{LL}$ with itself.

We now proceed with a series of definitions leading to the notion of self amalgamation.

Definition 11. (The dual of a consequence relation) Let $\mid\!\sim$ be a consequence relation in language **L**. Consider a dual language $\mathbf{L}^*$ defined as follows:

1. The atoms of $\mathbf{L}^*$ are all atoms of the form q^*, where q is an atom of **L**.
2. The connectives of $\mathbf{L}^*$ are all connectives of the form $\#^*$, where # is a connective of **L**.

Let * be a maping from **L** to $\mathbf{L}^*$ defined by

3. $(q)^* = q^*$, for q atomic
4. $(\#(A_1,\ldots,A_n))^* = \#^*(A^*_1,\ldots,A^*_n)$, for a connective #.
5. Define $\Delta^* \mid\!\sim^* A^*$ iff def $\Delta \mid\!\sim A$.

Example 13. In classical logic let $\wedge$ and $\vee$ be duals and let $A^* = \neg A$.

For further study of the ideas of this section, see [36].

4. A Case Study: What is the Logic of Classical Linear Logic

A lot has been said about linear logic. From our point of view, the system is very simple; it is the logic based on the data structures of multisets, satisfying the Deduction Theorem and fortified with additional structural and shift connectives. It is our purpose in this section to present linear logic from this point of view[4].

We start with the data structures of multisets for both the data and goal. Consider the language with $\rightarrow$ only and consider the smallest Scott type consequence relation $\mid\!\sim$ satisfying the deduction theorem for $\rightarrow$.

We know from example 3. that the smallest Tarski type consequence relation $\mid\!\sim_1$ for multisets satisfying the deduction theorem does characterise implicational intuitionistic linear logic. Thus it is clear that

$$\Delta \mid\!\sim_1 A \text{ iff } \Delta \mid\!\sim \{A\}.$$

What is not clear is the nature of the consequence $\Delta \mid\!\sim_2 A$ defined by

$$\Delta \mid\!\sim_2 A \text{ iff } \{A\} \mid\!\sim \Delta.$$

We shall address this problem later.

Given $\mid\!\sim$, let us add to the language the structural connectives $\oplus_1$, $\oplus_2$ and the shift connective $\odot$, as we did in the previous section, when we were discussing such connectives for an arbitrary consequence relation. We get for our case that the following holds, where $X \equiv Y$ means both $X \mid\!\sim Y$ and $Y \mid\!\sim X$:

1. $A^{\circ\circ} \equiv A$
2. $A \oplus_2 B \equiv A^{\circ} \rightarrow B$
3. $A \oplus_1 B \equiv (A \rightarrow B^{\circ})^{\circ}$
4. $(A \oplus_1 B)^{\circ} \equiv A^{\circ} \oplus_2 B^{\circ}$
5. $(A \oplus_2 B)^{\circ} \equiv A^{\circ} \oplus_1 B^{\circ}$

We can also add the "of course" connective $\boxed{!}$ and proof net markers. Let us agree that $\dagger_n\dagger$ is the corresponding connective on the right. We thus have:

6. $A \oplus_1 \boxed{!}_1 \equiv \boxed{!}_1 A$
7. $A \oplus_2 \boxed{!}_2 A \equiv \boxed{!}_2 A$
8. $A \oplus_1 B\dagger_n\dagger \mathrel{|\!\sim} C \oplus_2 (B^{\circ})\dagger_n\dagger$ is the same as $A \oplus_1 B \mathrel{|\!\sim} C$
9. $A \oplus_1 B^{\circ}\dagger_n\dagger \mathrel{|\!\sim} C \oplus_2 B\dagger_n\dagger$ is the same as $A \oplus_1 B^{\circ} \mathrel{|\!\sim} C$.

The moral of the story is that the properties of all of these connectives are determined by the structure and their geometric meaning.

The additives $\boxed{\wedge}$ and $\boxed{\vee}$ can be added to linear logic as in the example 11. When added to the language alongside $\oplus_1$ and $\oplus_2$, one can form formulas with arbitrary nestings of $\boxed{\wedge}$, $\boxed{\vee}$, $\oplus_1$, $\oplus_2$ within themselves. Given the meaning of these connectives, the corresponding meatalevel structures are the composition of two consequence relations, one for the additives over sets and one for the multiplicatives over multisets, as in Definition 8.

In the case of the additives of linear logic, the structures τ^* were multisets of sets (not hereditary). These can be regarded as a family of structures τ of $\mathcal{M}'$ (i.e. multisets) obtained by choosing all possible points from the sets of points in τ^*. For example if τ^* is ({a, b}, {c, d}, {e}) then the following set of τ's are associated.

(a, c, e), (a, d, e), (b, c, e) and (b, d, e).

We write formally $\tau \in \tau^*$ to indicate the connection. We can define the function δ^* to give values (wff) to each point. Thus databases and goals become lists of sets of formulas or multisets of sets of formulas.

For example, if Δ^* is ({A, B}, {C, D}, {E}) we get the following associated databases

(A, C, E), (A, D, E), (B, C, E) and (B, D, E).

We write again $\Delta \in \Delta^*$ to indicate the association.

Our goal is to define a consequence relation $\Delta^* \mathrel{|\!\sim}^* \Gamma^*$ by stipulating:

$\Delta^* \mathrel{|\!\sim}^* \Gamma^*$ iff $_{\text{def.}}$ for all $\Delta \in \Delta^*$, $\Gamma \in \Gamma^*$, $\Delta \mathrel{|\!\sim} \Gamma$.

We need to show that we get an S-consequence relation and for that we need to define the *Surgical Cut Rule*; we need to define substitution of one structure in another. We refer back to the general definition of the *composition* of two S-consequence relations.

We now present a theorem about the Hilbert formulation of linear logic with $\rightarrow$ and $\neg$ which shows that $\odot$ can be taken as $\neg$ and that $\neg$ can be mapped into the implicational fragment.

Theorem 2. Let **LL**($\neg$) be the extension of **LL** of example 3. with the unary symbol $\neg$ and the following axioms:

$\Vdash \neg\neg A \leftrightarrow A$
$\Vdash (\neg(A \rightarrow \neg B) \rightarrow C) \leftrightarrow (A \rightarrow (B \rightarrow C))$
$\Vdash (A \rightarrow B) \rightarrow (\neg B \rightarrow \neg A)$

Let $\bot$ be an arbitrary atom of **LL**. Let $\phi^{\bot}(A)$ be a translation from **LL**($\neg$) into **LL**($\bot$) defined as follows:

$\phi^{\bot}(q) = (q \rightarrow \bot) \rightarrow \bot$, q atomic
$\phi^{\bot}(A \rightarrow B) = \phi^{\bot}(A) \rightarrow \phi^{\bot}(B)$
$\phi^{\bot}(\neg A) = \phi^{\bot}(A) \rightarrow \bot$

Then the following holds:

LL($\neg$) $\Vdash$ A iff **LL** $\Vdash \phi^{\bot}(A)$.

Remark. The previous theorem yields the following:

1. Semantics for **LL**($\neg$), through the semantics for **LL**.
2. It shows that $\neg$, $\oplus_1$ and $\oplus_2$ (i.e. all the structural connectives of linear logic) are already definable from the impliction $\rightarrow$, via the "internal" translation $\phi^{\bot}$, for a fixed atom $\bot$.
3. It shows that $\neg$ can also be regarded as a negation, and not merely as a shift operator.

Example 14. (Cut Free Formulation of Linear Logic) In this section, we started with a consequence relation $\vdash_{LL}$ for multisets as data, added the obvious structural and shift connectives together with the additives, defined the symmetric amalgamation and got a consequence relation $\mid\!\sim$. Certain properties of $\mid\!\sim$ were listed in the previous example. It is possible to list enough properties of $\mid\!\sim$ to enable one to derive the *Cut Rule*. When this is done one gets a cut free formulation of the sytem $\mid\!\sim$. Although we have been referring to $\mid\!\sim$ as linear logic, we have not proved that $\mid\!\sim$ is indeed the system known in the literature as linear logic. To show that, all we need to do is to take a known formulation of linear logic, say $\vdash_1$, see e.g. [27], and prove that all $\vdash_1$ rules are valid in $\mid\!\sim$. This will show that $\vdash_1 \subseteq \mid\!\sim$. If we prove the Cut Elimination for $\vdash_1$ this will show that $\mid\!\sim \subseteq \vdash_1$. Thus essentially Cut Elimination for $\vdash_1$ establishes that $\mid\!\sim$ is indeed linear logic.

*Department of Computing
Imperial College of Science, Technology and Medicine, London

Notes

1 El presente artículo es una versión parcial del aparecido en P. Schroeder-Heister y K. Dosen (eds.): 1993, *Substructural Logics,* Oxford University Press,.

2 There are several versions of the **Cut Rule** in the literature, they are all equivalent for the cases of classical and intuitionistic logic but are not equivalent in the context of this paper. The version in the main text we call **Transitivity (Lemma Generation)**. Another version is:

$$\frac{\Gamma \mid\!\sim A \qquad \Delta, A \mid\!\sim B}{\Delta, \Gamma \mid\!\sim B}$$

This version implies **monotonicity**, when added to **Reflexivity**.
Another version we call **Internal Cut:**

$$\frac{\Delta, A \mid\!\sim \Gamma \qquad \Delta \mid\!\sim A, \Gamma}{\Delta \mid\!\sim \Gamma}$$

3 The perceptive reader may ask why we are doing all of this shifting. Instead of writing plainly (C, A) $\mid\!\sim$ B, for example, we code it as C $\mid\!\sim$ (B, $\circ$ A), putting A on the right and then *marking* it by $\circ$ to indicate that it should really be on the left. Then to confuse matters even further, we write A on the left anyway and mark it so that the duplication is cancelled, i.e. (C, $\dagger A \dagger$ $\mid\!\sim$ (B, $\circ \dagger A \dagger$).

The answer can be found in understanding the annotation as indicating *resource*. A structured database presents formulas to be used in the deduction in a way compatible with their structured layout. For linear logic the layout is a multiset and so each formula is to be used exactly once. In the course of the proof the formulas may be scattered about and/or duplicated. We need annotations to keep track of what is happening. This is why these connectives are useful.

4 The reader is cautioned that the phrase "from our point of view" is important. Intuitionistic logic, from our point of view, is essentially the smallest monotonic consequence relation with the Deduction Theorem for $\rightarrow$, with sets as structures and the structural and shift connectives. However, intuitionistic logic arose in a context of far greater importance and motivation in the historical development of logic. Similarly, linear logic has its own historical context and to understand its role one has to go back as early as 1976 [25]. For these reasons, Avron [26] should be viewed in context.

BIBLIOGRAPHY

[1] Gabbay, D.M.: 1985, 'Theoretical foundations for non-monotonic reasoning in expert systems', in Apt, K.R. (ed.): *Proceedings NATO Advanced Study Institute on Logics and Models of Concurrent Systems, LaColle-sur-Loup, France,* Berlin, Springer, 439-457.

[2] Makinson, D.: 1989, 'General theory of cumulative inference', in Reinfrank (ed.): *Proceedings Second International Workshop on Non-monotonic Reasoning,* Lecture Notes in Computer Science, Berlin, Springer, 1-18.

[3] Kraus, S., Lehmann, D. and Magidor, M.: 1990, 'Non monotonic reasoning, preferential models and cumulative logics', *Artificial Intelligence* 44, 167-207.

[4] Shoham, Y.: 'A semantical approach to non-monotonic logics', in *Proceedings Logics in Computer Science, (Ithaca, NY 1987),* 275-279.

[5] Shoham, Y.: 1988, *Reasoning about Change*, Cambridge, MA, MIT Press.

[6] Wojcicki, R.: 'An axiomatic treatment of non monotonic arguments', *Studia Logica*, to appear.

[7] Wojcicki, R.: 'Heuristic rules of inference in non-monotonic arguments', *Studia Logica*, to appear.

[8] Lehmann, D.: 'What does a conditional knowledge base entail?', in *KR 89, Toronto, May 89*, Morgan Kauffman Publisher.

[9] Lehmann, D. and Magidor, M.: 'Rational logics and their models: a study in cumulative logics', submitted for publication.

[10] Besnard, P.: 1989, 'Axiomatisation in the metatheory of non-monotonic inference system, *Proceedings CSCSI99*, Edmonton, 117-124.

[11] Akama, S.: 'Mathematical foundations of non-monotonic logics', *Journal of Automated Reasoning*, to appear.

[12] Makinson, D.: 'General patterns in nonmonotonic reasoning', a chapter in Gabbay, D.M., Hogger, C.J., J.A. Robinson (eds.): *Handbook of Logic in Artificial Intelligence and Logic Programming*, Volume 3, Oxford University Press, to appear.

[13] Gabbay, D.M.: *Labelled Deductive Systems*, 1st draft September 1989, 6th Draft Feb 91, to appear as a book with OUP.

[14] Vermeir, D. and Laenens, E.: 1990, 'An overview of ordered logic', in *Abstracts of the Third Logical Biennial*, Varna, Bulgaria.

[15] Nute, D.: 1986, *LDR -A Logic for Defeasible Reasoning*, ACMC Research Report 01-0013.

[16] Gabbay, D.M.: 1991, 'Algorithmic proof with diminishing resource, I', in E. Börger, E., Kleine Büning, H., Richter, M.M. and Schönfeld, W. (eds.): *Proceedings of CSL 90*, LNCS 533, Springer, 156-173.

[17] Wansing, H.: 1990, *Formulas as types for a hierarchy of sublogics of intuitionistic propositional logic*, Technical report no. 9/90, Free University of Berlin, 1-29.

[18] Gabbay, D.M.: 'Theoretical Foundations for non monotonic reasoning, Part 2: Structured non-monotonic Theories', in *SCAI '91, Proceedings of the Third Scandanavian Conference on AI*, Amsterdam, IOS Press, 19-40.

[19] Belnap, N.: 1982, 'Display Logic', *Journal of Philosophical Logic* 11, 375-417.

[20] Gabbay, D.M.: 1981, *Semantical Investigations in Heyting's Intuitionistic Logic*, D. Reidel.

[21] Lauritzen, S.L. and Spiegelhalter, D.J.: 'Local Computation with Probabilities on Graphical Structures and their Application to Expert Systems (with discussion)', *J Roy Statist Soc B* 50, 157-224.

[22] Alchourron, C.E., Gärdenfors, P. and Makinson, D.: 1985, 'On the logic of theory change: Partial meet contraction and revision function', *The Journal of Symbolic Logic* 50, 510-530.

[23] Gärdenfors, P.: 1988, *Knowledge in Flux: Modelling the Dynamics of Epistemic States*, Cambridge, MA, The MIT Press, Bradford Books.

[24] Makinson, D. and Gärdenfors, P.: 'Relations between the logic of theory change and nonmonotonic logic', in Brewka, G. and Freitag, H. (eds.): *Arbeitspapier der GMD no 443: Proceedings of the Workshop on Nonmonotonic reasoning*, 7-27. Also in Fuhrmann, A. and Morreau, M. (eds.): 1990, *The logic of theory change*, Berlin, Springer Verlag, Lecture Notes in Artificial Intelligence no. 465, 185-205.

[25] Girard, J.-Y.: 1976, '- 3 valued logic and cut elimination: The actual meaning of Takeuti's conjectures', *Dissertationes Mat* CXXXVI.

[26] Avron, A.: 1988, 'The semantics and Proof theory of linear logic', *Theoretical Computer Science* 57, 161-184.

[27] Girard, J.-Y.: 1987, 'Linear Logic', *Theoretical Computer Science* 50, 1-102.

[28] Abramsky, S.: 1990, *Computational Interpretation of Linear Logic*, Research Report DoC 90/20, London, Department of Computing, Imperial College.

[29] Ryan, M: 'Default and Revision in Structured Theories', in *Proceedings of LICS 91*, 362-373.

[30] Crocco, G. and Farinas del Cerro, L.: *Structural Logics*, Draft April 91.

[31] Gabbay, D.M.: 1992, 'Theory of Algorithmic Proof', a Chapter in Abramsky, S., Gabbay, D. and Maibaum, T. (eds.): *Handbook of Logic in Theoretical Computer Science*, volume 2, OUP, 311-413.

[32] van Benthem, J.: 1988, 'The Lambek Calculus', in Oerhrle, R. et al. (ed.): *Categorical Grammars and Natural Language Structures*, Reidel, 35-68.

[33] Urquhart, A.: 1972, 'Semantics for Relevant Logics', *Journal of Symbolic Logic* 37, 159-169.

[34] Gabbay, D.M.: *Fibred Semantics and the Weaving of Logics*, Manuscript, Imperial College, September 1992.

[35] Gabbay, D.M.: 1992, 'How to construct a logic for your application', in *Proceedings of the 16th German AI Conference, GWAI 92*, Springer Lecture notes on AI, vol. 671, 1-30.

[36] Wansing, H.: *Tarskian structured consequence relation*, to appear.

ESTUDIOS