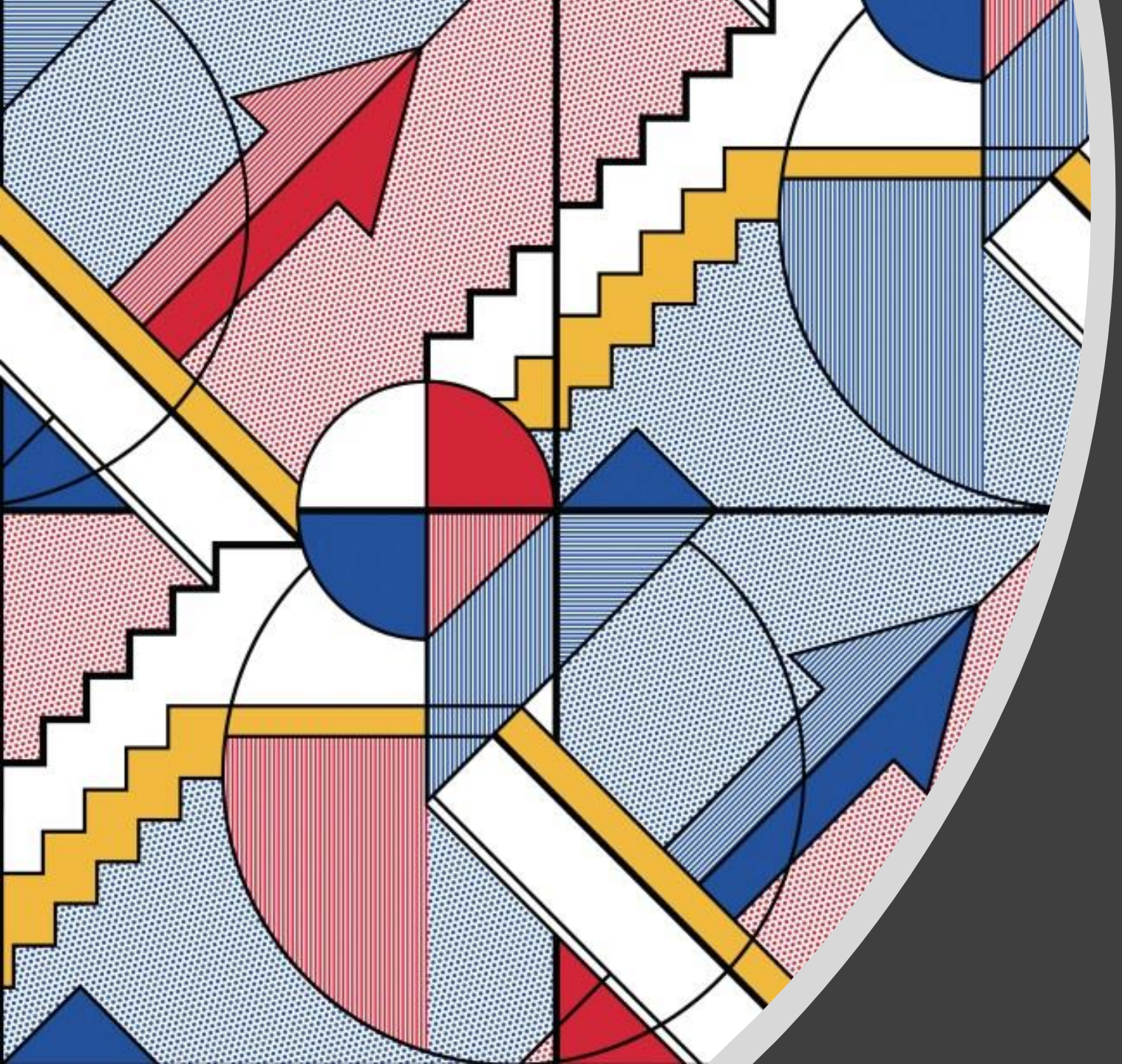
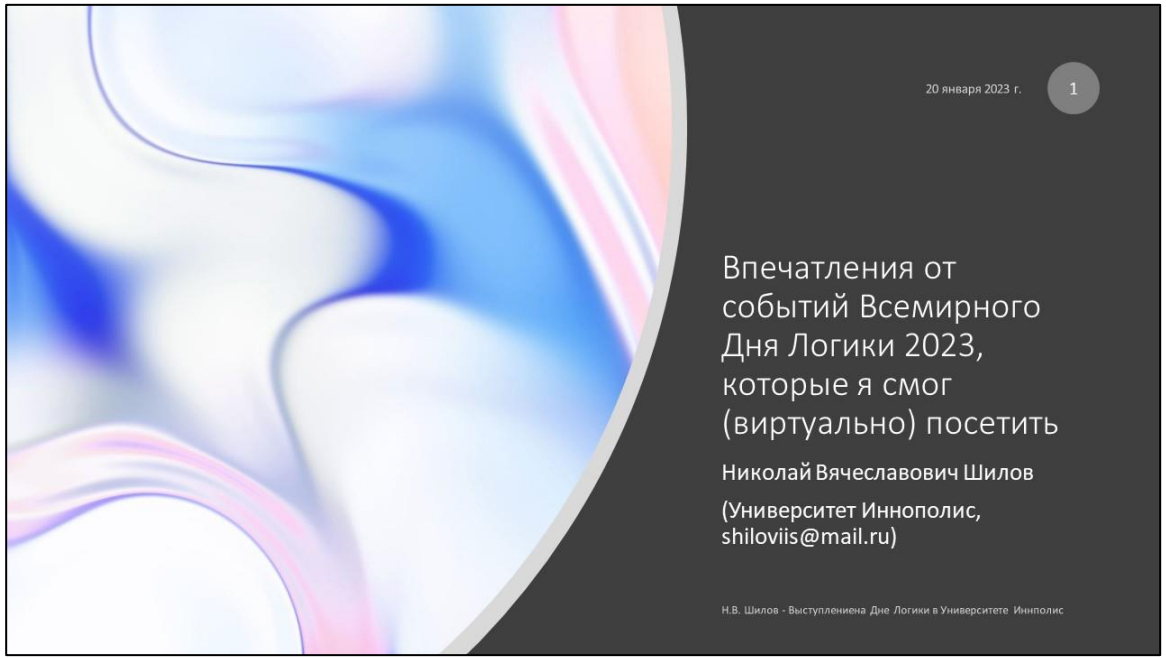


Введение в субструктурные логики (в продолжение Дня Логики 2023)

Николай Вячеславович Шилов

(Университет Иннополис, shiloviis@mail.ru)

В продолжение Дня Логики 2023



13 января: Юбилейное заседание научно-исследовательского семинара «Формальная философия» (НИУ ВШЭ)

https://llfp.hse.ru/news/808354837.html, https://llfp.hse.ru/formphil_seminar

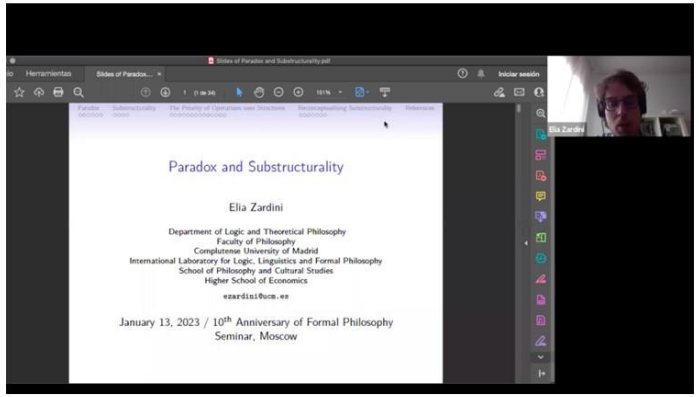
- *В своем докладе Э. Дзардини, отталкиваясь от ряда ключевых парадоксов, обсуждаемых в рамках философии логики (парадоксы лжеца, материальной импликации и др.), представил основные субструктурные подходы к рассмотрению парадокса и отметил преимущества данных подходов перед классическим рассмотрением, опирающимся на аппарат традиционных структурных логик.*

Драгалина-Черная Елена Григорьевна
заведующая Международная лаборатория логики, лингвистики и формальной философии

- Есть запись доклада: https://youtu.be/ll6hUPL76V4

13 января: Юбилейное заседание научно-исследовательского семинара «Формальная философия» (НИУ ВШЭ)

https://llfp.hse.ru/news/808354837.html, https://llfp.hse.ru/formphil_seminar



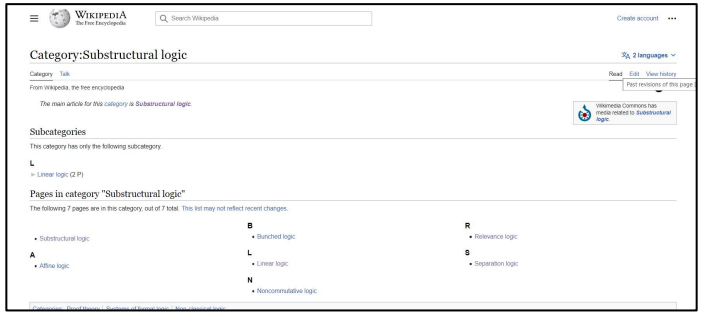
Аксиомы Плагиатора
(Источник: Андрей Александрович Берс, 26 июля 1934, Свердловск — 28 января 2013, Новосибирск)

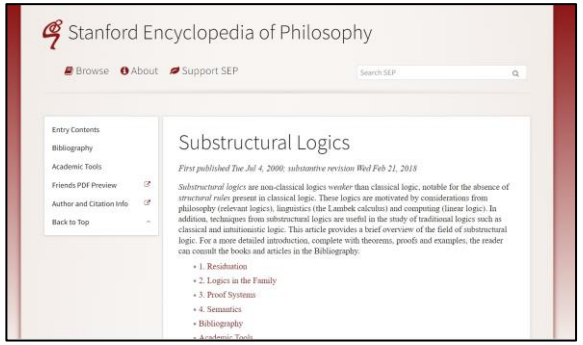
Если ты списываешь

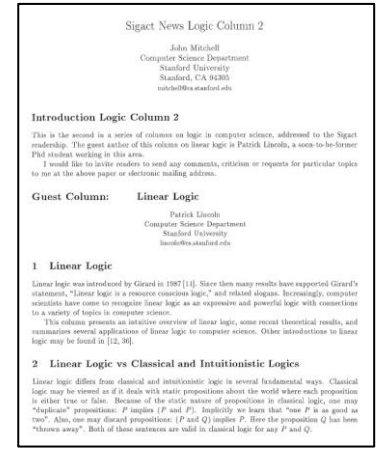
- с одного источника – это плагиат
- с двух – это компиляция
- с трех – это научный обзор

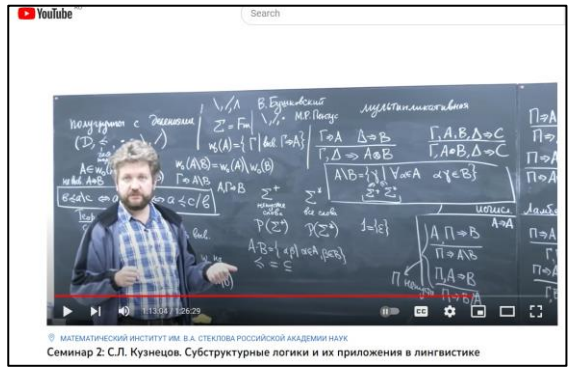
(А еще есть аксиомы жизни в малометражной квартире)

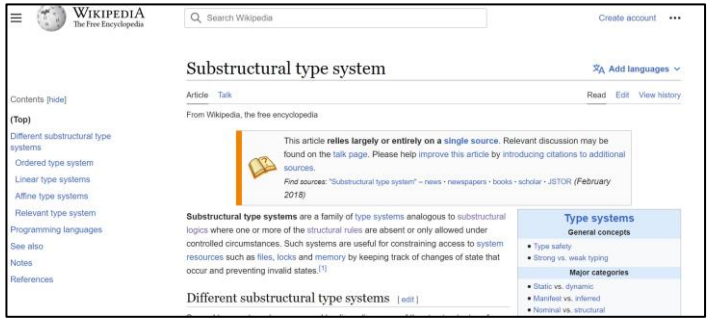
Если с трех – это научный обзор



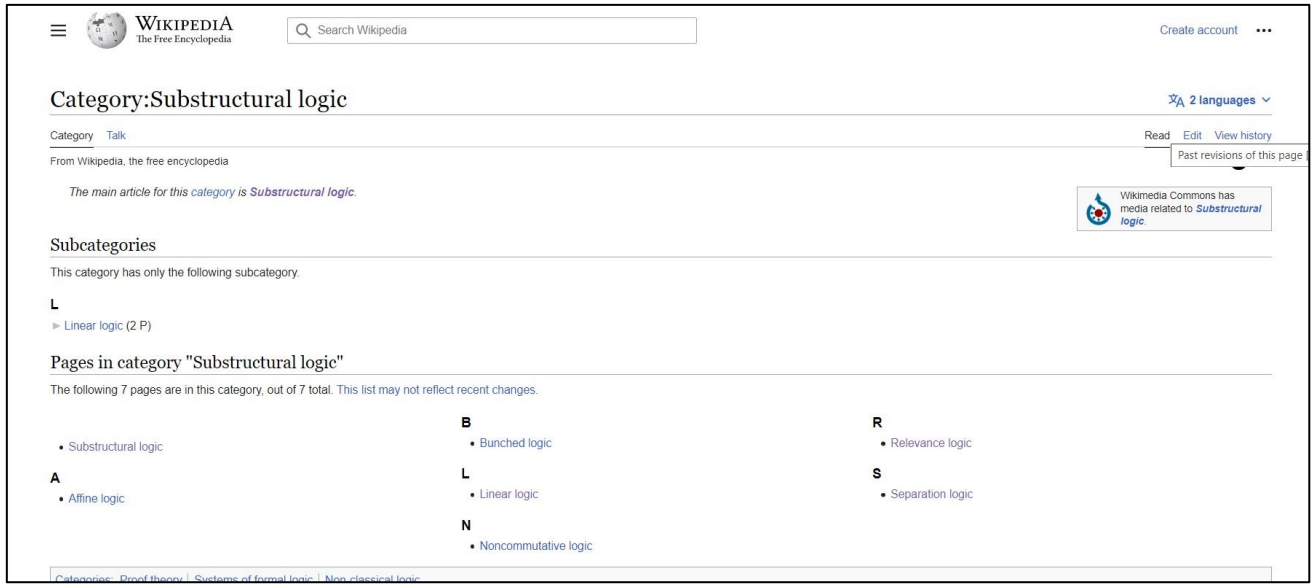








Category: Substructural logic (Wikipedia)



WIKIPEDIA
The Free Encyclopedia

Search Wikipedia

Create account

Category:Substructural logic

2 languages

Category Talk

Read Edit View history

Past revisions of this page

From Wikipedia, the free encyclopedia

*The main article for this category is **Substructural logic**.*

Wikimedia Commons has media related to ***Substructural logic***.

Subcategories

This category has only the following subcategory.

L

- Linear logic (2 P)

Pages in category "Substructural logic"

The following 7 pages are in this category, out of 7 total. This list may not reflect recent changes.

- Substructural logic

A

- Affine logic

B

- Bunched logic

L

- Linear logic

N

- Noncommutative logic

R

- Relevance logic

S

- Separation logic

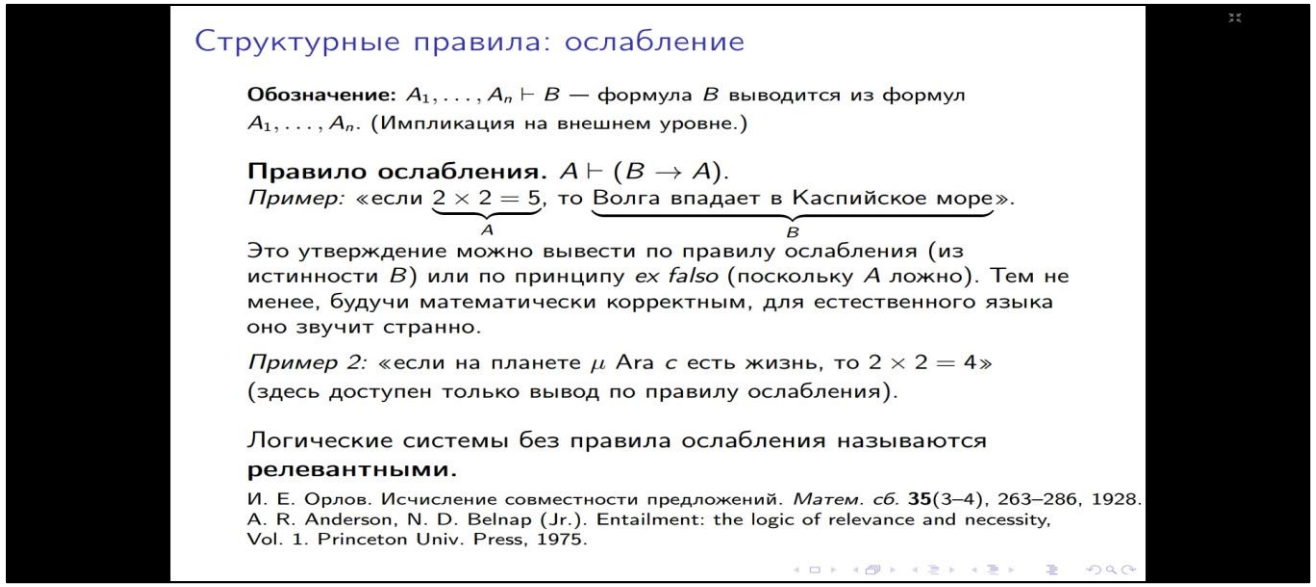
Categories: Proof theory | Systems of formal logic | Non-classical logic

Субструктурные логики — от алгебры до лингвистики

С. Л. Кузнецов

Зимняя школа «Симметрия и сложность в математике», матфак ВШЭ, январь 2019

лекции 1–4, 1–2 февраля 2019 г.



Структурные правила: ослабление

Обозначение: $A_1, \ldots, A_n \vdash B$ — формула B выводится из формул $A_1, \ldots, A_n$. (Импликация на внешнем уровне.)

Правило ослабления. $A \vdash (B \to A)$.

Пример: «если $\underbrace{2 \times 2 = 5}_{A}$, то $\underbrace{\text{Волга впадает в Каспийское море}}_{B}$».

Это утверждение можно вывести по правилу ослабления (из истинности B) или по принципу *ex falso* (поскольку A ложно). Тем не менее, будучи математически корректным, для естественного языка оно звучит странно.

Пример 2: «если на планете μ Ara c есть жизнь, то $2 \times 2 = 4$» (здесь доступен только вывод по правилу ослабления).

Логические системы без правила ослабления называются **релевантными.**

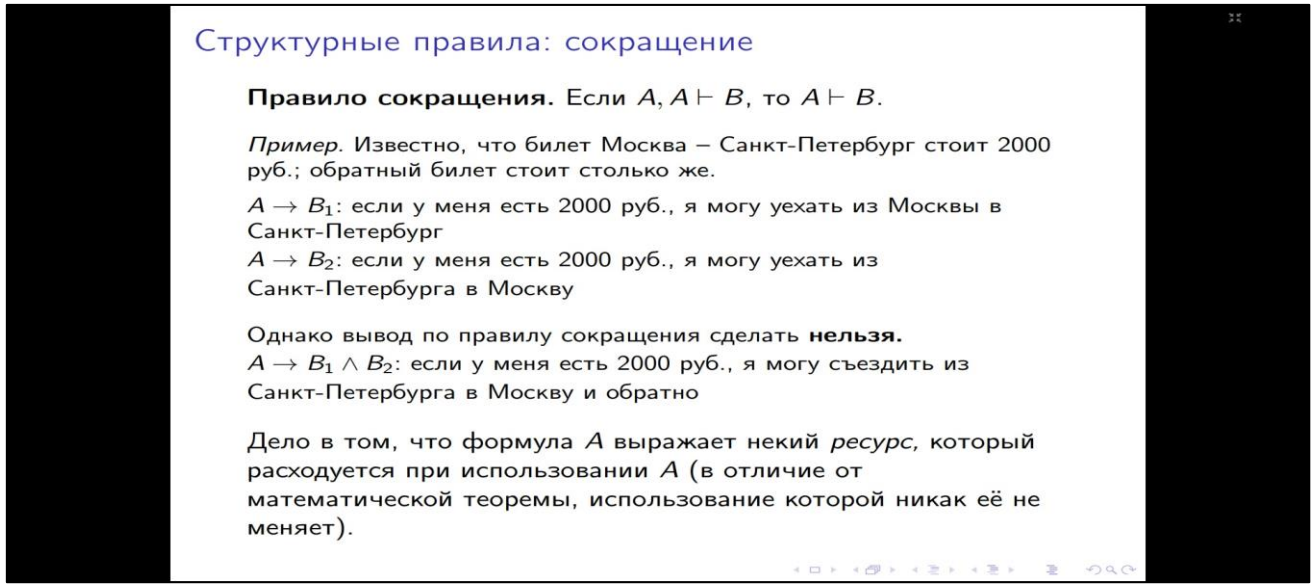
И. Е. Орлов. Исчисление совместности предложений. *Матем. сб.* **35**(3–4), 263–286, 1928.
A. R. Anderson, N. D. Belnap (Jr.). Entailment: the logic of relevance and necessity, Vol. 1. Princeton Univ. Press, 1975.

Субструктурные логики — от алгебры до лингвистики

С. Л. Кузнецов

Зимняя школа «Симметрия и сложность в математике», матфак ВШЭ, январь 2019

лекции 1–4, 1–2 февраля 2019 г.



Структурные правила: сокращение

Правило сокращения. Если $A, A \vdash B$, то $A \vdash B$.

Пример. Известно, что билет Москва – Санкт-Петербург стоит 2000 руб.; обратный билет стоит столько же.

$A \to B_1$: если у меня есть 2000 руб., я могу уехать из Москвы в Санкт-Петербург

$A \to B_2$: если у меня есть 2000 руб., я могу уехать из Санкт-Петербурга в Москву

Однако вывод по правилу сокращения сделать **нельзя.**

$A \to B_1 \wedge B_2$: если у меня есть 2000 руб., я могу съездить из Санкт-Петербурга в Москву и обратно

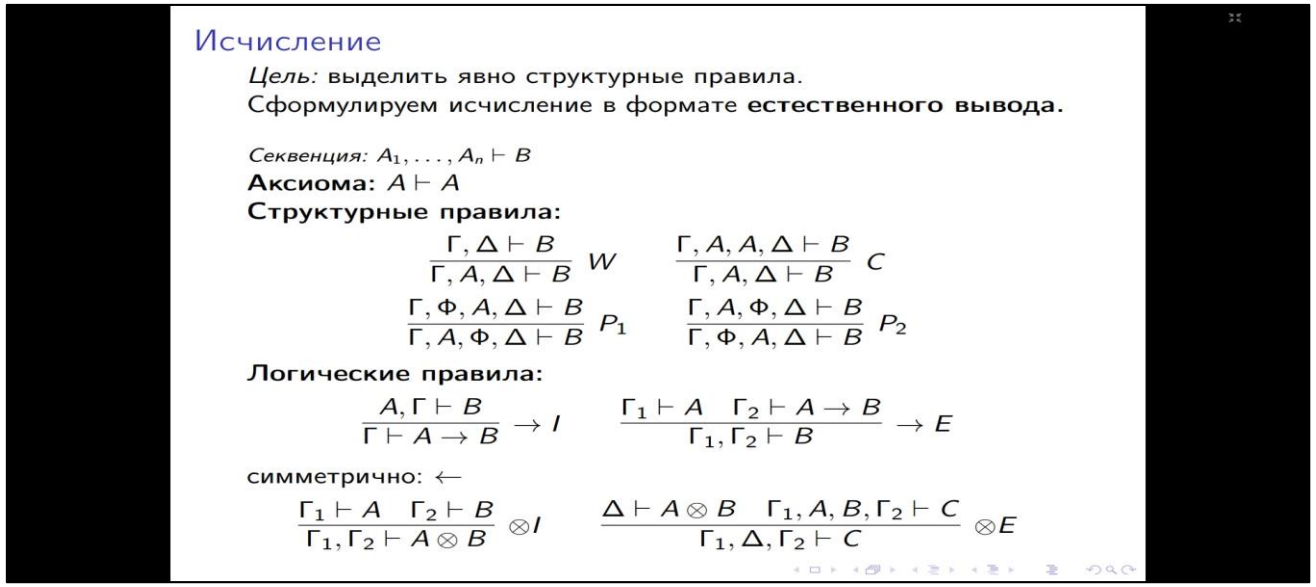
Дело в том, что формула A выражает некий *ресурс*, который расходуется при использовании A (в отличие от математической теоремы, использование которой никак её не меняет).

Субструктурные логики — от алгебры до лингвистики

С. Л. Кузнецов

Зимняя школа «Симметрия и сложность в математике», матфак ВШЭ, январь 2019

лекции 1–4, 1–2 февраля 2019 г.



Исчисление

Цель: выделить явно структурные правила.

Сформулируем исчисление в формате **естественного вывода.**

Секвенция: $A_1, \ldots, A_n \vdash B$

Аксиома: $A \vdash A$

Структурные правила:

$$\frac{\Gamma, \Delta \vdash B}{\Gamma, A, \Delta \vdash B}\, W \qquad \frac{\Gamma, A, A, \Delta \vdash B}{\Gamma, A, \Delta \vdash B}\, C$$

$$\frac{\Gamma, \Phi, A, \Delta \vdash B}{\Gamma, A, \Phi, \Delta \vdash B}\, P_1 \qquad \frac{\Gamma, A, \Phi, \Delta \vdash B}{\Gamma, \Phi, A, \Delta \vdash B}\, P_2$$

Логические правила:

$$\frac{A, \Gamma \vdash B}{\Gamma \vdash A \to B} \to I \qquad \frac{\Gamma_1 \vdash A \quad \Gamma_2 \vdash A \to B}{\Gamma_1, \Gamma_2 \vdash B} \to E$$

симметрично: $\leftarrow$

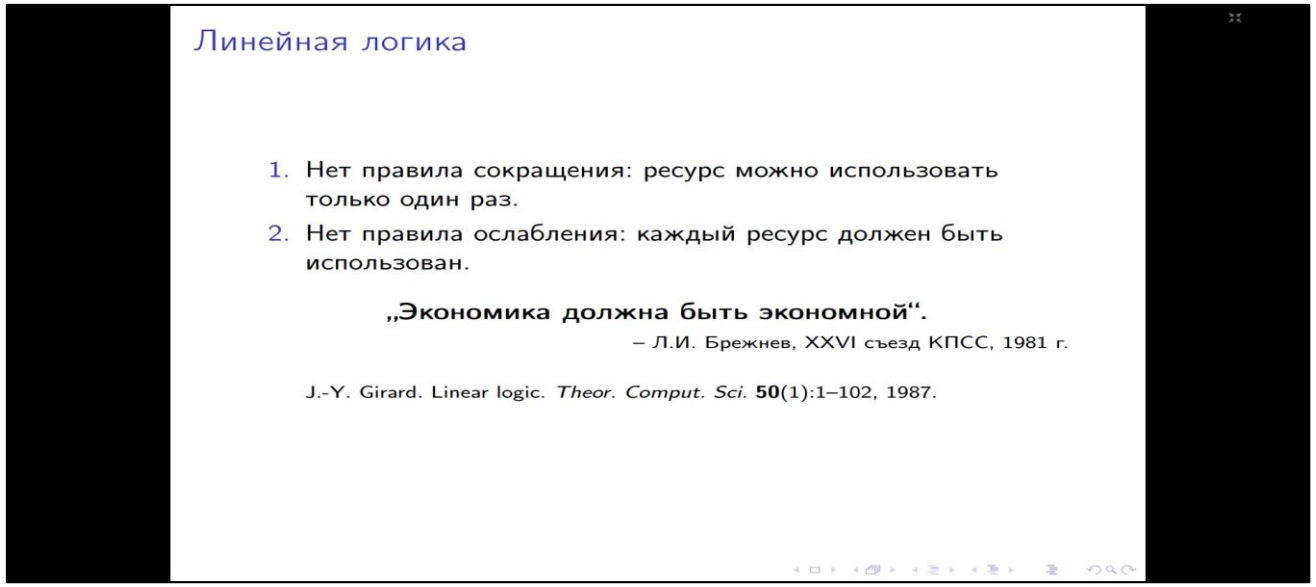
$$\frac{\Gamma_1 \vdash A \quad \Gamma_2 \vdash B}{\Gamma_1, \Gamma_2 \vdash A \otimes B} \otimes I \qquad \frac{\Delta \vdash A \otimes B \quad \Gamma_1, A, B, \Gamma_2 \vdash C}{\Gamma_1, \Delta, \Gamma_2 \vdash C} \otimes E$$

Субструктурные логики — от алгебры до лингвистики

С. Л. Кузнецов

Зимняя школа «Симметрия и сложность в математике», матфак ВШЭ, январь 2019

лекции 1–4, 1–2 февраля 2019 г.



Линейная логика

1. Нет правила сокращения: ресурс можно использовать только один раз.
2. Нет правила ослабления: каждый ресурс должен быть использован.

„Экономика должна быть экономной“.

– Л.И. Брежнев, XXVI съезд КПСС, 1981 г.

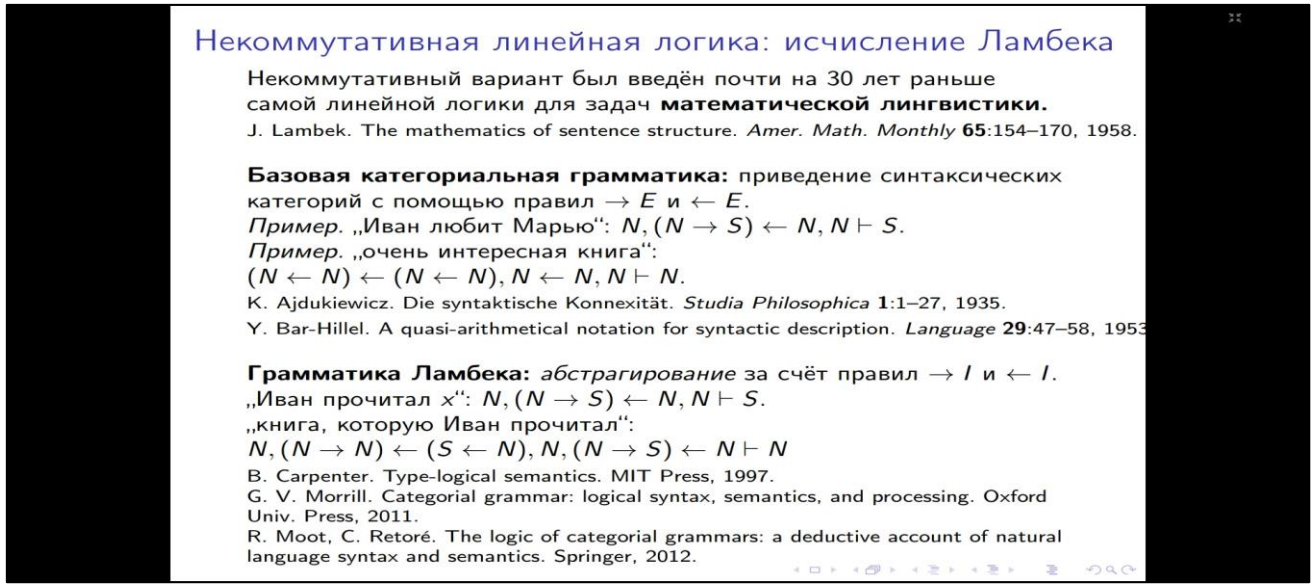
J.-Y. Girard. Linear logic. *Theor. Comput. Sci.* **50**(1):1–102, 1987.

Субструктурные логики — от алгебры до лингвистики

С. Л. Кузнецов

Зимняя школа «Симметрия и сложность в математике», матфак ВШЭ, январь 2019

лекции 1–4, 1–2 февраля 2019 г.



Некоммутативная линейная логика: исчисление Ламбека

Некоммутативный вариант был введён почти на 30 лет раньше самой линейной логики для задач **математической лингвистики.**

J. Lambek. The mathematics of sentence structure. *Amer. Math. Monthly* **65**:154–170, 1958.

Базовая категориальная грамматика: приведение синтаксических категорий с помощью правил $\rightarrow E$ и $\leftarrow E$.

Пример. „Иван любит Марью“: $N, (N \rightarrow S) \leftarrow N, N \vdash S$.

Пример. „очень интересная книга“:

$(N \leftarrow N) \leftarrow (N \leftarrow N), N \leftarrow N, N \vdash N$.

K. Ajdukiewicz. Die syntaktische Konnexität. *Studia Philosophica* **1**:1–27, 1935.

Y. Bar-Hillel. A quasi-arithmetical notation for syntactic description. *Language* **29**:47–58, 1953

Грамматика Ламбека: *абстрагирование* за счёт правил $\rightarrow I$ и $\leftarrow I$.

„Иван прочитал x“: $N, (N \rightarrow S) \leftarrow N, N \vdash S$.

„книга, которую Иван прочитал“:

$N, (N \rightarrow N) \leftarrow (S \leftarrow N), N, (N \rightarrow S) \leftarrow N \vdash N$

B. Carpenter. Type-logical semantics. MIT Press, 1997.

G. V. Morrill. Categorial grammar: logical syntax, semantics, and processing. Oxford Univ. Press, 2011.

R. Moot, C. Retoré. The logic of categorial grammars: a deductive account of natural language syntax and semantics. Springer, 2012.

Logics in the Family
(Stanford Encyclopedia of Philosophy)

2.1 Relevant Logics

Many people have wanted to give an account of logical validity which pays some attention to conditions of *relevance*. If $X, A \vdash B$ holds, then X must somehow be *relevant* to A. Premise combination is restricted in the following way. We may have $X \vdash A$ without also having $X, Y \vdash A$. The new material Y might not be relevant to the deduction. In the 1950s, Moh (1950), Church (1951) and Ackermann (1956) all gave accounts of what a 'relevant' logic could be. The ideas have been developed by a stream of workers centred around Anderson and Belnap, their students Dunn and Meyer, and many others. The canonical references for the area are Anderson, Belnap and Dunn's two-volume *Entailment* (1975 and 1992). Other introductions can be found in Read's *Relevant Logic*, Dunn and Restall's *Relevance Logic* (2002), and Mares' *Relevant Logic*: *a philosophical interpretation* (2004).

Logics in the Family
(Stanford Encyclopedia of Philosophy)

2.2 Resource Consciousness

This is not the only way to restrict premise combination. Girard (1987) introduced *linear logic* as a model for processes and resource use. The idea in this account of deduction is that resources must be used (so premise combination satisfies the relevance criterion) and they do not extend *indefinitely*. Premises cannot be *re*-used. So, I might have $X, X \vdash A$, which says that I can use X twice to get A. I might not have $X \vdash A$, which says that I can use X once alone to get A. A helpful introduction to linear logic is given in Troelstra's *Lectures on Linear Logic* (1992). There are other formal logics in which the *contraction rule* (from $X, X \vdash A$ to $X \vdash A$) is absent. Most famous among these are Łukasiewicz's many-valued logics. There has been a sustained interest in logics without this rule because of Curry's paradox (Curry 1977, Geach 1995; see also Restall 1994 in Other Internet Resources).

Парадокс Карри (https://ru.wikipedia.org/wiki/Парадокс_Карри)

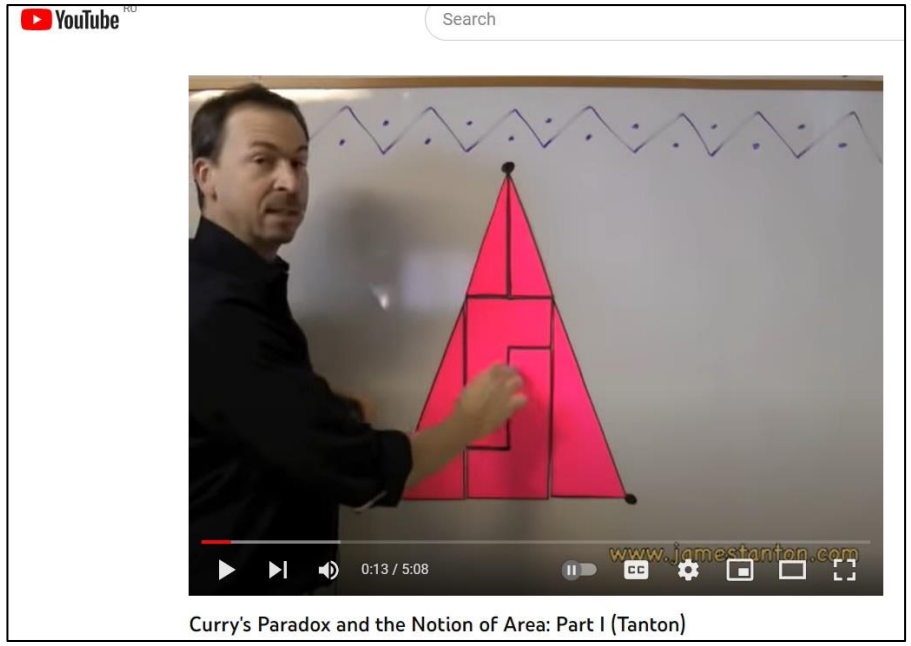
Парадокс Карри — парадоксальный вывод из утверждения: «Если это утверждение верно, то русалки существуют». Вместо существования русалок может указываться любое неправдоподобное или ложное заявление (в английском оригинале — существование Санта-Клауса). Ход мыслей, ведущий к парадоксу, строится следующим образом:

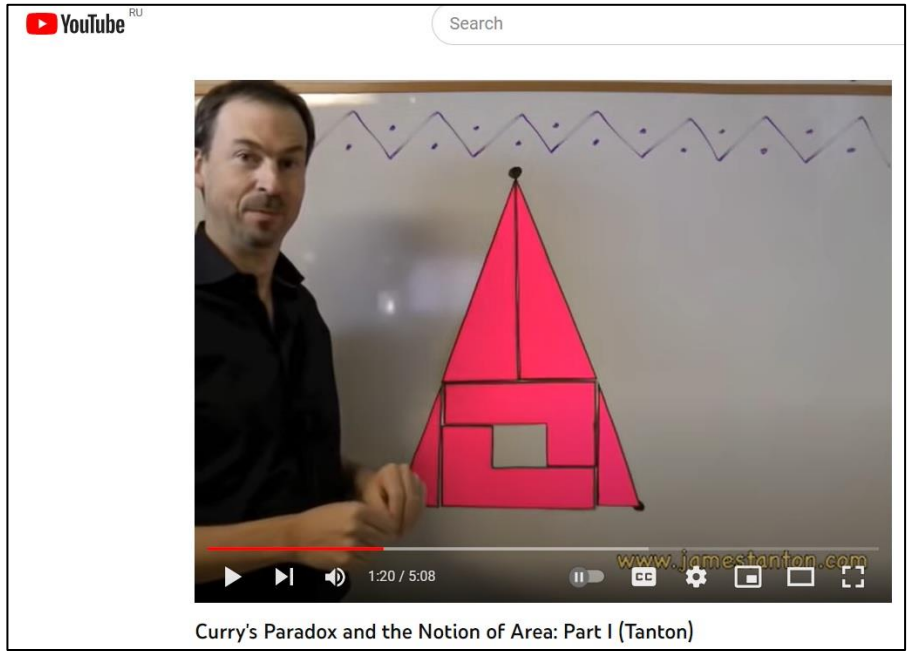
- Обозначим через S высказывание «Если S верно, то русалки существуют»;
- Мы не знаем, верно ли высказывание S. Но если бы высказывание S было верным, то это влекло бы существование русалок;
- Но именно это и утверждается в высказывании S, таким образом S — верно;
- Следовательно, русалки существуют!

Причиной парадокса Карри является использование в утверждении недопустимой ссылки на само себя. В строго формализованных теориях парадокс Карри не появляется, однако некоторые исследователи отмечают, что теорема Лёба может рассматриваться как результат формализации рассуждений, аналогичных парадоксу Карри, с помощью гёделевской нумерации.

Парадокс рассматривался математиком Хаскеллом Карри, в честь которого и получил своё название. Иногда называется парадоксом Лёба по имени Мартина Хуго Лёба.

Кстати: совсем другой парадокс совсем другого Карри (https://www.youtube.com/watch?v=eFw0878Ig-A)





Proof Systems
(Stanford Encyclopedia of Philosophy)

Gentzen systems, with their introduction rules on the left and the right, have very special properties which are useful in studying logics. Since connectives are always *introduced* in a proof (read from top to bottom) proofs never *lose* structure. If a connective does not appear in the conclusion of a proof, it will not appear in the proof at all, since connectives cannot be eliminated.

In certain substructural logics, such as linear logic and the Lambek calculus, and in the fragment of the relevant logic **R** without disjunction, a Gentzen system can be used to show that the logic is *decidable*, in that an algorithm can be found to determine whether or not an argument $X \vdash A$ is valid. This is done by searching for proofs of $X \vdash A$ in a Gentzen system. Since premises of this conclusion must feature no language not in this conclusion, and they have no greater complexity (in these systems), there are only a finite number of possible premises. An algorithm can check if these satisfy the rules of the system, and proceed to look for premises for these, or to quit if we hit an axiom. In this way, decidability of some substructural logics is assured.

Model Theory
(Stanford Encyclopedia of Philosophy)

While the relevant logic $\mathbf{R}$ has a proof system more complex than the substructural logics such as linear logic, which lack distribution of (extensional) conjunction over disjunction, its *model theory* is altogether more simple. A Routley-Meyer *model* for the relevant logic $\mathbf{R}$ is comprised of a set of *points* P with a three-place relation R on P. A conditional $A \to B$ is evaluated at a world as follows:

$A \to B$ is true at x if and only if for each y and z where $Rxyz$, if A is true at y, B is true at z.

An argument is *valid* in a model just when in any point at which the premises are true, so is the conclusion. The argument $A \vdash B \to B$ is invalid because we may have a point x at which A is true, but at which $B \to B$ is not. We can have $B \to B$ fail to be true at x simply by having $Rxyz$ where B is true at y but not at z.

Model Theory
(Stanford Encyclopedia of Philosophy)

It is one thing to use a semantics as a formal device to model a logic. It is another to use a semantics as an *interpretive* device to *apply* a logic. The literature on substructural logics provides us with a number of different ways that the ternary relational semantics can be applied to describe the logical structure of some phenomena in which the traditional structural rules do not apply.

For logics like the Lambek calculus, the interpretation of the semantics is straightforward. We can take the points to be linguistic items, and the ternary relation to be the relation of concatenation ($Rxyz$ if and only if x concatenated with y results in z). In these models, the structural rules of contraction, weakening and permutation all fail, but premise combination is associative.

The contemporary literature on linguistic classification extends the basic Lambek Calculus with richer forms of combination, in which more syntactic features can be modelled (see Moortgat 1995).

Model Theory
(Stanford Encyclopedia of Philosophy)

Another application of these models is in the treatment of the semantics of *function application*. We can think of the points in a model structure as both *functions* and *data*, and hold that $Rxyz$ if and only if x (considered as a function) applied to y (considered as data) is z. Traditional accounts of functions do not encourage this dual use, since functions are taken to be of a 'higher' than their inputs or outputs (on the traditional set-theoretic model of functions, a function *is* the set of its *input-output* pairs, and so, it can never take *itself* as an input, since sets cannot contain themselves as members). However, systems of functions modelled by the untyped λ-calculus, for example, allow for self-application. Given this reading of points in a model, a point is of type $A \to B$ just if whenever it takes inputs of type A, it takes outputs of type B. The inference rules of this system are then principles governing types of functions: the sequent

$$(A \to B) \,\&\, (A \to C) \vdash A \to (B \,\&\, C)$$

tells us that whenever a function takes As to Bs and As to Cs, then it takes As to things that are both B and C.

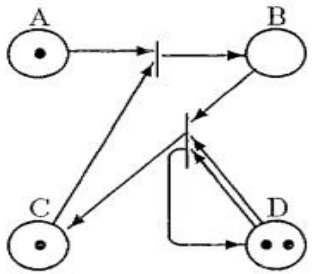
This example gives us a model in which the appropriate substructural logic is extremely weak. *None* of the usual structural rules (not even associativity) are satisfied in this model. This example of a ternary relational model is discussed in (Restall 2000, Chapter 11).

Liner Logic

Patrick Lincoln

5 Connections to Computer Science

There has been much recent excitement about linear logic in the logic-based theoretical computer science community. Most of this excitement stems from the newfound ability to capture difficult "resource" problems logically. For example, linear logic provides a natural and simple encoding of Petri net reachability. In linear logic the formula $!((\mathbf{a} \otimes \mathbf{c}) \multimap \mathbf{b})$ may be used to encode a Petri net transition taking tokens from place **a** and **c** and adding a token to place **b**. Similarly, the formula $!((\mathbf{b} \otimes \mathbf{d} \otimes \mathbf{d}) \multimap (\mathbf{c} \otimes \mathbf{d}))$ may be seen as a transition taking one token from **b** and two tokens from **d**, and adding one token to **c**. These transitions are presented graphically below:



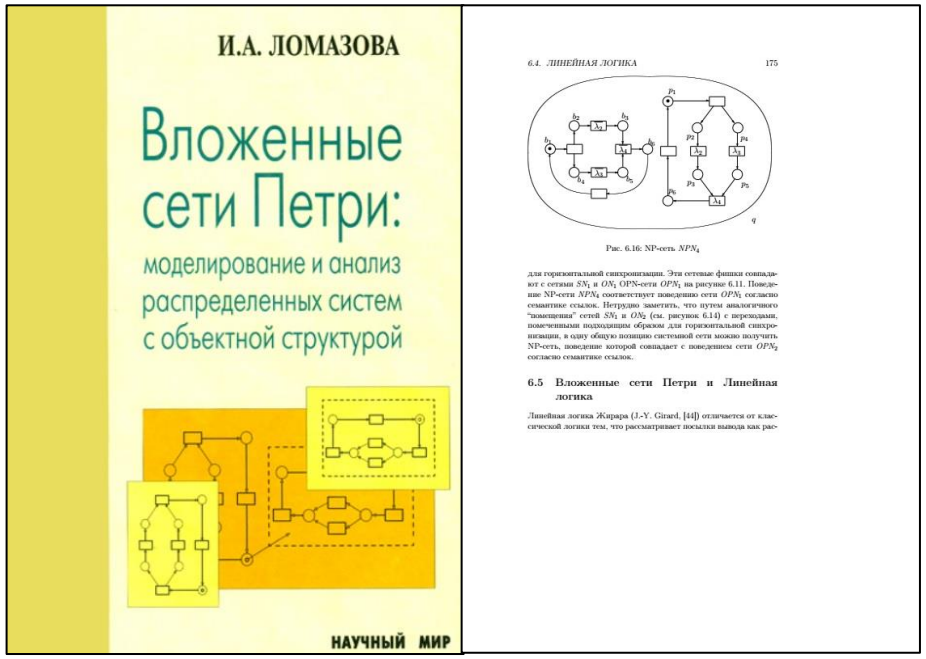
$$!((\mathbf{a} \otimes \mathbf{c}) \multimap \mathbf{b}),$$
$$!((\mathbf{b} \otimes \mathbf{d} \otimes \mathbf{d}) \multimap (\mathbf{c} \otimes \mathbf{d})),$$
$$\mathbf{a}, \mathbf{c}, \mathbf{d}, \mathbf{d}$$

Thus one can encode Petri net transitions as reusable linear implications. Tokens are represented as atomic propositions, and a reachability problem may be presented as a sequent:

$$!((\mathbf{a} \otimes \mathbf{c}) \multimap \mathbf{b}), !((\mathbf{b} \otimes \mathbf{d} \otimes \mathbf{d}) \multimap (\mathbf{c} \otimes \mathbf{d})), \mathbf{a}, \mathbf{c}, \mathbf{d}, \mathbf{d} \vdash \mathbf{c}, \mathbf{d}$$

This sequent is provable in linear logic if and only if there is a sequence of Petri net rule applications that transform the token set $\{a, c, d, d\}$ to $\{c, d\}$. This connection has been well-studied [5, 14, 30, 6, 9], and extended to cover other models of concurrency [22, 2, 35].

Вложенные сети Петри
и Линейная логика
Ломазова И. А.



Substructural type system (https://en.wikipedia.org/wiki/Substructural_type_system)

Substructural type systems are a family of type systems analogous to substructural logics where one or more of the structural rules are absent or only allowed under controlled circumstances. Such systems are useful for constraining access to system resources such as files, locks and memory by keeping track of changes of state that occur and preventing invalid states.[1]

Different substructural type systems [edit]

Several type systems have emerged by discarding some of the structural rules of exchange, weakening, and contraction:

	Exchange	Weakening	Contraction	Use
Ordered	—	—	—	Exactly once in order
Linear	Allowed	—	—	Exactly once
Affine	Allowed	Allowed	—	At most once
Relevant	Allowed	—	Allowed	At least once
Normal	Allowed	Allowed	Allowed	Arbitrarily

- **Ordered type systems** (discard exchange, weakening and contraction): Every variable is used exactly once in the order it was introduced.
- **Linear type systems** (allow exchange, but neither weakening nor contraction): Every variable is used exactly once.
- **Affine type systems** (allow exchange and weakening, but not contraction): Every variable is used at most once.
- **Relevant type systems** (allow exchange and contraction, but not weakening): Every variable is used at least once.
- **Normal type systems** (allow exchange, weakening and contraction): Every variable may be used arbitrarily.

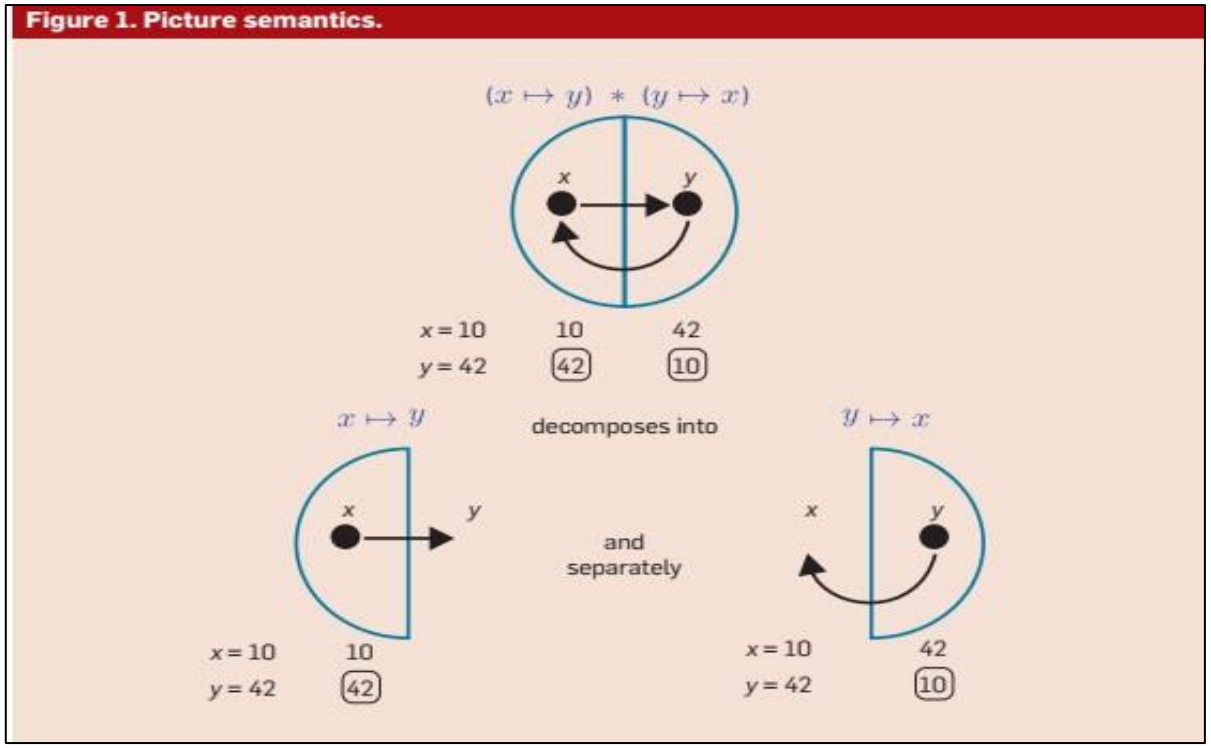
The explanation for affine type systems is best understood if rephrased as “every *occurrence* of a variable is used at most once”.

Programming languages [edit]

The following programming languages support linear or affine types:

- C++
- ATS
- Clean
- Idris
- Mercury
- F*
- LinearML
- Alms
- Haskell with GHC 9.0.1 or above.[1]
- Granule
- Rust
- Nim

Separation Logic
Peter O'Hearn



Separation Logic

Peter O'Hearn

Figure 2. Mathematical semantics.

- Assume a partial commutative monoid $(H, \circ, e)$, where $\circ : H \times H \rightharpoonup H$ and $e \in H$. Pre/post assertions denote elements of the powerset $\mathcal{P}(H)$.
- $*$ lifts $\circ$ to the powerset $\mathcal{P}(H)$: $P * Q$ is

  $$\{h_P \circ h_Q \mid h_P \circ h_Q \text{ defined and } h_P \in P \text{ and } h_Q \in Q\}$$

- emp denotes the singleton set of the empty heaplet: $\{e\}$.
- $-\!\!*$ is an implication quantifying over separate heaps: $P -\!\!* Q$ is

  $$\{h \mid \forall h_P.\ h \circ h_P \text{ defined and } h_P \in P \text{ implies } h \circ h_P \in Q\}$$

- In the RAM model H is the set of finite partial functions from positive integers (addressible locations) to integers, $h \circ h'$ is the union of functions with disjoint domain, and undefined when h and h' overlap. e is the empty partial function. The assertion $n \mapsto m$ denotes the singleton set $\{f\}$ where f maps n to m and is undefined elsewhere.
- To deal with variables and also quantifiers consider functions s from variables to integers, and extend the above semantics pointwise to pairs (s, h).

Separation Logic
Peter O'Hearn

Figure 3. Separation logic proof system (a selection).

SMALL AXIOMS

Pointer Write (Store)

$\{x \mapsto -\}[x] = v\ \{x \mapsto v\}$

Pointer Read (Load)

$\{x \mapsto v\}y = [x]\ \{y == v \wedge x \mapsto v\}$

Allocation

$\{emp\}x = \texttt{alloc}()\ \{x \mapsto -\}$

De-Allocation

$\{x \mapsto -\}\texttt{free}(x)\ \{emp\}$

LOCAL REASONING RULES

Frame Rule

$$\frac{\{pre\}code\ \{post\}}{\{pre * frame\}code\ \{post * frame\}}$$

Concurrency Rule

$$\frac{\{pre_1\}process_1\ \{post_1\} \quad \{pre_2\}process_2\ \{post_2\}}{\{pre_1 * pre_2\}process_1 \parallel process_2\ \{post_1 * post_2\}}$$