

# **Онтологическая инженерия вычислений: к материалистической парадигме семантики языков программирования**

Научный руководитель: \_\_\_\_\_

Научный рецензент: \_\_\_\_\_

Научный оппонент: \_\_\_\_\_

Автор: Шипков В. И.

**Москва-2026**

## Оглавление

Аннотация.....	3
1. Введение.....	5
1.1. Онтологический разрыв в современном программировании: к постановке проблемы.....	5
1.2. Термодинамический предел абстракции.....	5
1.3. Степень разработанности проблемы.....	6
1.4. Цель и задачи исследования.....	7
2. Методологические основания онтологической инженерии вычислений.....	8
2.1. Диалектический материализм как методология системного анализа.....	8
2.2. Термодинамическая модель вычисления.....	8
2.3. Формализация противоречий как систем ограничений.....	9
2.4. Линейная логика как ресурсная онтология.....	11
2.5. Социально-экономическое измерение: когнитивная стоимость.....	11
2.6. Синтез: архитектура ресурсно-контрактной семантики.....	12
3. Формальная модель ресурсно-контрактной семантики языка «Контекст».....	14
3.1. Онтологические категории.....	14
3.2. Синтаксис.....	14
3.3. Вычисление ресурсного следа.....	16
3.4. Примеры.....	17
3.5. Связывание и запрет.....	18
3.6. Операционная семантика.....	19
3.7. Пример программы.....	20
Библиографический список.....	21

## Аннотация

**Цель:** Разработка формальной методологии проектирования семантики языков программирования (ЯП), основанной на онтологической редукции вычислительных абстракций к физическому субстрату. Создание инструментальной базы для преодоления идеалистической парадигмы в программировании через внедрение категорий материального бытия (ресурс, энергия, противоречие) в ядро типовых систем.

**Методы:** Междисциплинарный синтез методов системного анализа (2.3.1), философской онтологии (5.7.6) и термодинамики вычислений. Используются: диалектический системный подход к формализации противоречий; аппарат модальной логики ресурсов; теория информационной энтропии (физика); методология социально-экономического моделирования когнитивных издержек разработки. Предложена концепция «ресурсно-контрактной семантики», где вычислительный процесс моделируется как физическое движение с конечными энергетическими и информационными затратами.

**Результаты:** Впервые построена формальная модель семантики ЯП, изоморфная термодинамике реальных вычислительных систем (с явным выделением энтропийных и энергетических параметров). Обоснована корректность разрешения системных противоречий (гибкость/производительность, стоимость/надежность) через реконфигурацию ресурсных контрактов на различных этапах проектирования и исполнения. В отличие от существующих подходов, где ресурсные ограничения являются внешними мета-свойствами, в предлагаемом подходе они конституируют (constitute) семантику на уровне типовой системы, что реализует принцип материалистической редукции абстракций к физическому субстрату. Разработан прототип языка «Контекст», демонстрирующий возможность материалистической редукции абстракций (отказ от аксиомы бесконечной памяти и бесконечного времени) и социальной адаптации инструмента (снижение когнитивного барьера через «раскрывающуюся сложность»).

**Значимость:** Результаты создают основания для нового направления в системном анализе — онтологической инженерии, интегрирующей физические законы, социальные практики разработки и философские основания технологий. Предложенный подход позволяет снизить экзистенциальную отчужденность разработчиков от материальной природы вычислений и повысить надежность сложных систем через учет термодинамических ограничений на этапе как компиляции, так и исполнения.

**Ключевые слова:** системный анализ; онтология программирования; диалектический материализм; ресурсно-контрактная семантика; термодинамика вычислений; формальная редукция абстракций; социальная инженерия ПО; когнитивная экономика разработки.

**Коды специальностей ВАК:** 2.3.1 (Системный анализ, управление и обработка информации); 5.7.6 (Философия науки и техники); 1.2.2 (Математическое моделирование, численные методы и комплексы программ).

**Для рецензента (примечание к подаче):** данная статья является междисциплинарной и адресована научному сообществу в областях системного анализа (формализация ресурсных ограничений), философии науки (онтология технических систем) и социологии разработки ПО (когнитивные аспекты инструментария).

# 1. Введение

## 1.1. Онтологический разрыв в современном программировании: к постановке проблемы

Современная индустрия программного обеспечения (ПО) переживает системный кризис, корни которого лежат не в технической, но в методологической (а следовательно, онтологической) плоскости. Несмотря на экспоненциальный рост вычислительной мощности, наблюдаемый «разрыв семантики» [1] между спецификацией программы и её физическим поведением приводит к нарастанию технического долга, энергетическому кризису центров обработки данных [2] и когнитивной перегрузке разработчиков [3].

Традиционные формальные семантики (денотационная, операционная, аксиоматическая) базируются на платонической онтологии, постулирующей существование «чистых» математических объектов, не зависящих от физического субстрата [4]. Это проявляется в фундаментальных аксиомах:

- бесконечная лента Тьюринга [5],
- нулевая стоимость  $\beta$ -редукции в  $\lambda$ -исчислении [6],
- отсутствие времени как физического параметра в логике Хоара [7].

Данная установка отражает идеалистическую парадигму в программировании, где абстрактная форма признаётся первичной по отношению к материальному содержанию (ресурсам энергии, времени, пространства), что порождает отчуждение [8] разработчиков от физической природы вычислений.

Диалектический материализм позволяет иначе концептуализировать вычисление: как отражение объективной реальности в субъективной модели, как процесс движения материи (носителей информации), подчинённый законам термодинамики [9, 10]. В рамках данной парадигмы программа не существует «априори» в мире идей, а возникает только как процесс взаимодействия физических систем (кристаллы памяти, электронные потоки, тепловое излучение), что требует онтологической редукции семантических категорий к категориям материального бытия.

## 1.2. Термодинамический предел абстракции

Физика вычислений устанавливает фундаментальные ограничения, игнорируемые классической семантикой. Принцип Ландауэра [11] и теорема Беннетта [12] демонстрируют, что логически необратимые операции (стирание информации) сопряжены с диссипацией энергии

$$E \geq kT \ln 2,$$

а предельная плотность вычислений ограничена энтропийными барьерами [13]. Однако существующие языки программирования (ЯП) трактуют память и время как бесконечные ресурсы, что приводит к антропогенному конфликту между растущей сложностью ПО и физическими законами сохранения.

Системный анализ [14] показывает, что данное противоречие носит структурный характер: абстракции высокого уровня (объекты, функции, монады) скрывают энтропийные издержки, создавая иллюзию «бесплатной» выразительности. Это порождает экзистенциальный разрыв между семантикой языка (что декларируется) и онтологией вычисления (что происходит физически), что в социальном измерении ведёт к формированию «касты технологических жрецов» [15], владеющих эзотерическими знаниями о скрытой материальной природе системы, и «профанов» (прикладных разработчиков), лишённых понимания ресурсных ограничений.

### 1.3. Степень разработанности проблемы

Попытки преодоления онтологического разрыва предпринимались в нескольких направлениях.

В физике вычислений исследованы термодинамические пределы (Ландауер [11], Маргулос [16]), однако они не были интегрированы в практику языков программирования как онтологические примитивы.

В формальной логике линейная логика Жирара [17] ввела ресурсную интерпретацию (формулы как consumable resources), что стало теоретической основой для систем владения (Rust [18], LinearML [19]). Однако данные подходы ограничиваются управлением памятью, игнорируя энергетические и социально-экономические измерения ресурса.

В системном анализе разработаны методы учёта ограничений (constraints satisfaction) и теория системной динамики (Forrester [20]), но они применяются к программированию как внешние регуляторы, а не как внутренняя семантика языка.

В философии техники обсуждалась проблема «отчуждения» в цифровых системах (Хайдеггер [21], Фуко [22], современные авторы [23]), однако без инструментализации в формальных методах разработки.

Таким образом, существует научный пробел: отсутствие формальной методологии проектирования семантики ЯП, в которой физические ограничения (энергия, энтропия, пространство-время) и социальные параметры (когнитивная стоимость, доступность) были бы конститутивными (constitutive), а не внешними ограничениями; где диалектика противоречий

(тезис–антитезис–синтез) формализована как механизм адаптации; где «материалистическая редукция» абстракций к субстрату реализована на уровне типовой системы.

## 1.4. Цель и задачи исследования

Цель данной работы — разработка методологии онтологической инженерии вычислений, основанной на синтезе диалектического материализма (как методологии анализа противоречий), термодинамики информационных процессов и формальной семантики, направленной на создание языка программирования, семантика которого изоморфна физической реальности вычислений.

### **Задачи:**

- Формализация категорий диалектического материализма (противоречие, становление, отражение) в рамках системного анализа как иерархических систем ресурсных ограничений с динамическим разрешением конфликтов.
- Разработка ресурсно-контрактной семантики, интегрирующей термодинамические параметры (энергетическая стоимость, энтропийный след) в типовую систему языка.
- Обоснование корректности механизма разрешения противоречий между декларативностью и эффективностью через количественный переход в качественный (снижение уровня строгости при критическом дефиците ресурсов).
- Создание прототипа языка «Контекст» с реализацией социальной адаптации инструмента (принцип «раскрывающейся сложности») и экспериментальная проверка снижения когнитивной нагрузки и плотности дефектов.

## 2. Методологические основания онтологической инженерии вычислений

### 2.1. Диалектический материализм как методология системного анализа

Диалектический материализм предоставляет онтологические категории, применимые к анализу сложных систем, включая вычислительные процессы. В рамках данной работы используются три фундаментальные категории, формализуемые в контексте проектирования семантики ЯП [9, 14]:

**Противоречие** (contradiction) — не как логическая противоположность, но как **динамическое взаимодействие сторон объекта, порождающее его развитие**. В системном анализе противоречие формализуется как система ограничений

$$C = \{c_1, c_2, \dots, c_n\},$$

не допускающая общего решения в статической постановке (несовместная система неравенств). Разрешение противоречия есть не отрицание одной стороны, а **синтез** — построение динамического оператора перехода между состояниями, где конфликтующие требования реализуются в разных фазах или масштабах процесса.

**Становление** (becoming) — процесс непрерывного изменения качества объекта при количественном накоплении изменений. Формализуется через **динамические системы с качественными переходами** (bifurcations): при достижении критического значения параметра  $p \geq p_{crit}$  система переключается между режимами функционирования, сохраняя идентичность через непрерывность трансформации.

**Отражение** (reflection) — воспроизведение структуры объективной реальности в субъективной модели. В онтологической инженерии — **изоморфизм** между семантическими конструкциями языка и физическими процессами вычисления, исключающий онтологический разрыв (semantic gap) между спецификацией и реализацией.

Данные категории конституируют **материалистическую онтологию вычислений**, альтернативную платонической: программа существует не как абстрактная сущность в «мире идей», но как **процесс движения материи** (электронов, фотонов, спинов), подчинённый законам сохранения и термодинамики [10, 11].

### 2.2. Термодинамическая модель вычисления



Физический субстрат вычисления характеризуется тремя фундаментальными параметрами, игнорируемыми классической семантикой:

**Энергетическая стоимость** ( $E$ ) — работа, необходимая для выполнения операции. Согласно принципу Ландауэра [10], стирание одного бита информации требует энергии:

$$E_{\text{erase}} \geq k_B T \ln 2$$

где:

-  $k_B$  — постоянная Больцмана,

-  $T$  — температура среды.

Логически обратимые вычисления (Bennett [12]) позволяют асимптотически приблизиться к нулевой энергетической стоимости, но требуют пространственных ресурсов (памяти для сохранения промежуточных состояний).

**Энтропийный след** ( $S$ ) — мера необратимости вычислительного процесса. Рост энтропии системы при вычислении:

$$\Delta S \geq \frac{\Delta Q}{T}$$

связывает информационную обработку с тепловыделением, что ограничивает плотность вычислений (Lloyd [13]).

**Временная протяжённость** ( $\tau$ ) — физическое время выполнения, связанное с пространственной конфигурацией вычислителя через предел Марголуса—Левитина [16]:

$$\tau \geq \frac{\pi \hbar}{2 E}$$

В рамках предлагаемой методологии данные параметры становятся **первичными** (first-class) в семантике языка: любая абстракция (функция, объект, тип) несёт явный **ресурсный след**  $\rho = (E, S, \tau)$ , измеримый в физических единицах.

## 2.3. Формализация противоречий как систем ограничений

Противоречия в вычислительных системах носят структурный характер и поддаются формализации через аппарат **иерархических систем ограничений** (hierarchical constraint systems) [14].

**Определение 1.**

*Ресурсный контракт* — кортеж:

$$\mathcal{K} = \langle R_{avail}, R_{req}, P, \delta \rangle,$$

где:

- $R_{avail} \in \mathbb{R}_+^3$  — вектор доступных ресурсов (энергия  $E$ , энтропия  $S$ , время  $t$ );
- $R_{req} \in \mathbb{R}_+^3$  — вектор требуемых ресурсов для выполнения операции;
- $P \in [0, 1]^3$  — веса приоритетов конфликтующих требований;
- $\delta : \mathbb{R}_+^3 \times \mathbb{R}_+^3 \rightarrow \{0, 1\}$  — предикат выполнимости:

$$\delta(R_a, R_r) = 1 \Leftrightarrow P \odot R_{avail} \geq P \odot R_{req}$$

(покомпонентно взвешенное доминирование, где  $\odot$  — поэлементное произведение).

## Определение 2.

*Противоречие* — состояние  $\mathcal{K}$ , при котором  $\delta(\mathcal{K}) = 0$ .

В идеалистической парадигме противоречие разрешается через **исключение** (exsertion). В материалистической — через **диалектический синтез**: система переконфигурируется путём снижения уровня абстракции  $A \downarrow A'$  или ослабления строгости проверок  $S \downarrow S'$  до достижения  $\delta(\mathcal{K}) = 1$ .

## Определение 3.

*Оператор эволюции*  $\mathcal{U}_{\Delta t} : \mathcal{K}_{adm} \rightarrow \mathcal{K}$ , где

$\mathcal{K}_{adm} = \{\mathcal{K} \in \mathcal{K} \mid \exists \mathcal{K}' : \delta(\mathcal{K}') = 1 \wedge Core(\mathcal{K}') = Core(\mathcal{K})\}$  задаётся как:

$$\mathcal{U}_{\Delta t}(\mathcal{K}) = \arg \min_{\mathcal{K}' \in \mathcal{N}(\mathcal{K})} \mathcal{F}(\mathcal{K}')$$

где  $\mathcal{N}(\mathcal{K})$  — окрестность  $\mathcal{K}$  в метрике ресурсных контрактов  $d(\mathcal{K}_1, \mathcal{K}_2) = \|P \odot (R_{a1} - R_{a2})\|_1$ , а  $\mathcal{F}(\mathcal{K}') = E' - T \cdot S'$  — **свободная энергия** системы (аналог термодинамического потенциала), минимизация которой обеспечивает устойчивость при сохранении функциональной корректности ядра  $Core(\mathcal{K})$ .

**Теорема 1 (Существование синтеза).** Если  $\mathcal{K} \in \mathcal{K}_{adm}$ , то  $\mathcal{U}_{\Delta t}(\mathcal{K})$  сходится за конечное число итераций.

*Доказательство.* Множество  $\mathcal{N}(\mathcal{K})$  конечно (дискретные уровни абстракции). Функция  $\mathcal{F}$  ограничена снизу (энергия неотрицательна). При каждой итерации  $\mathcal{F}$  строго убывает или достигается  $\delta=1$ . Следовательно, процесс завершается.  $\square$

## 2.4. Линейная логика как ресурсная онтология

Линейная логика Жирара [17] предоставляет формальный аппарат для моделирования **материальности** логических операций. В классической логике аксиома тавтологии  $A \rightarrow A$  допускает бесконечное копирование и стирание формул; в линейной логике формулы — **ресурсы**, подлежащие расходованию:

- **Мультипликативная конъюнкция**  $(A \otimes B)$  — комбинация ресурсов, требующая наличия и  $A$ , и  $B$ .
- **Линейная импликация**  $(A \multimap B)$  — процесс преобразования ресурса  $A$  в ресурс  $B$  (consumable transformation).
- **Модальность**  $!A$  (of course) — возможность репликации ресурса с явной стоимостью копирования  $C_{copy}(A)$ .

В контексте онтологической инженерии линейная логика интерпретируется как **исчисление ресурсных потоков**. Тип в языке «Контекст» — линейная формула, доказательство которой в ресурсном исчислении соответствует корректному вычислительному процессу с балансом ресурсов.

Связь с термодинамикой: правило вывода  $A \vdash A$  (идемпотентность) в классической логике соответствует обратимому процессу с  $\Delta S=0$ ; в линейной логике оно запрещено, что отражает необратимость реальных физических операций.

## 2.5. Социально-экономическое измерение: когнитивная стоимость

Материализм требует учёта не только физических, но и **социальных** форм движения материи. Разработка ПО — труд, подчинённый законам экономики [8, 22].

Введём **когнитивную стоимость** ( $C_{cog}$ ) — энергию (в нейрофизиологическом и экономическом смысле), затрачиваемую разработчиком на освоение и применение инструментария. Данный параметр подчиняется **закону убывающей отдачи**:

$$\frac{\partial Productivity}{\partial Complexity} > 0 \text{ при } Complexity < \Theta_{crit}$$

$$\frac{\partial Productivity}{\partial Complexity} < 0 \text{ при } Complexity > \Theta_{crit}$$

где  $\Theta_{crit}$  — критический порог когнитивной нагрузки.

Противоречие между **мощностью** (выразительностью) и **доступностью** (педагогичностью) языка формализуется как ресурсный контракт:

$$\mathcal{K}_{ped} = \langle C_{max}^{cog}, Expressiveness, \lambda, \delta_{learn} \rangle$$

где  $\lambda$  — кривая обучения (функция усвоения компетенций от времени),  $\delta_{learn}$  — предикант достижения порога профессиональной пригодности.

Разрешение — **принцип раскрывающейся сложности** (progressive disclosure): семантика языка стратифицирована на уровни  $L_1 \subset L_2 \subset \dots \subset L_n$ , и переход  $L_i \rightarrow L_{i+1}$  происходит только при накоплении необходимой компетенции (количественный переход в качественное понимание). Формально:

$$\mathcal{U}_{learn}: (L_i, Competence) \rightarrow \begin{cases} L_i & \text{если } Competence < \Theta_i \\ L_{i+1} & \text{иначе} \end{cases}$$

## 2.6. Синтез: архитектура ресурсно-контрактной семантики

Предлагаемая методология интегрирует перечисленные компоненты в единую онтологическую рамку:

Категория диалектики	Формализация	Реализация в языке
Противоречие	Несовместная система ограничений $\mathcal{K}$	Кризис контракта + триггер реконфигурации $\mathcal{U}_{\Delta t}$
Становление	Динамический оператор с бифуркацией	Runtime-адаптация уровня строгости $A \downarrow A'$

Отражение	Изоморфизм семантика ↔ термодинамика	Ресурсные аннотации типов $\rho=(E,S,\tau)$
Материя	Ресурсный след $\rho$	Измеримые параметры операций
Социальное	Когнитивная стоимость $C_{cog}$ C	Градуированная система обучения $\mathcal{U}_{learn}$

Данная архитектура обеспечивает **материалистическую редукцию**: любая абстракция языка (функция высшего порядка, полиморфный тип, монада) раскладывается (reduces) на конечный набор ресурсных параметров, поддающихся физическому измерению и социальной оценке.

### 3. Формальная модель ресурсно-контрактной семантики языка «Контекст»

#### 3.1. Онтологические категории

Категория	Сущность	Роль
Протон	Данные с ресурсным следом	Содержимое, материал
Нейтрон	Действие с энергетической стоимостью	Операция над протонами
Атом	Объединение протонов и нейтронов	Составная структура (опциональное)
Кварк	Базовая материальная единица	Фиксированный набор, самостоятельное использование запрещено

(// Кварк  $\in$  {Булево, Байт, Целое, Вещ, Строка})

(// Кварки самостоятельно использовать запрещено — только в составе протонов и нейтронов)

**Допустимые кварки языка:**

Кварк	Ресурсный след (время, память, энергия)
Булево	(1, 1, 1)
Байт	(1, 1, 1)
Целое	(10, 8, 20)
Вещ	(15, 16, 40)
Строка	(5+2n, 2+n, 10+3n) (// n — длина)

Все атрибуты ресурсов задаются как **конкретные значения Целое**.

#### 3.2. Синтаксис

**Контекст:**

контекст ::= ( детерминатор тело )  
детерминатор ::= имя-протона | имя-нейтрона | имя-атома  
тело ::= последовательность | конструктор | вызов | управление

**Конструктор кварка** (только внутри протона или нейтрона):

(Кварк значение)  
(// Кварк € {Булево, Байт, Целое, Вещ, Строка})  
(// Самостоятельное использование (Кварк значение) вне протона –  
ошибка)

#### **Нейтрон с атрибутами:**

```
(нейтрон ИмяНейтрона(  
    параметр ПротонПараметра  
    ...  
)ПротонРезультата  
@время Целое  
@память Целое  
@энергия Целое  
(  
    телоНейтрона  
    (вернуть рез)  
)  
)
```

#### **Протон с атрибутами:**

```
(протон ИмяПротона(  
    часть1 Протон1  
    часть2 Протон2  
    ...  
)  
@время Целое  
@память Целое  
@энергия Целое  
(  
    (// тело опционально)  
)  
)
```

#### **Атом с атрибутами:**

```
(атом ИмяАтома(  
    протон1 Протон1  
    протон2 Протон2  
    ...  
    нейтрон1 Нейтрон1  
    нейтрон2 Нейтрон2
```

```

    ...
)
@время Целое (// Целое)
@память Целое (// Целое)
@энергия Целое (// Целое)
(
    (// инициализация)
)
)

```

**Связывание имени:**

```
(контекст)(уст имяПеременной)
```

**Запрет связывания:**

```
(контекст)()
(// () – результат запрещён и недоступен)
```

### 3.3. Вычисление ресурсного следа

Ресурс любой сущности — **сумма:**

- собственных атрибутов @время, @память, @энергия
- ресурса конструирования из кварков
- ресурса применения нейтронов

**Пример вычисления:**

```

(нейтрон Сложить(
    а Целое
    б Целое
)Целое
@время 5
@память 4
@энергия 10
(
    (Целое сложить а б)
    (вернуть рез)
)
)

```



)

Ресурс вызова `(x Сложить y)` где `x`, `y` — Целое:

- время:  $5 + 10 + 10$  (// атрибут + два кварка)
- память:  $4 + 8 + 8$
- энергия:  $10 + 20 + 20$

### 3.4. Примеры

**Простые кварки** (только внутри протонов):

```
(протон Сто(  
  @время 10  
  @память 8  
  @энергия 20  
  (  
    (Целое 100)  
  )  
) (уст сто)  
  
(протон Приветствие(  
  @время 17  
  @память 9  
  @энергия 34  
  (  
    (Строка "привет")  
  )  
) (уст текст)
```

(// Неправильно: (Целое 100)(уст сто) — кварк вне протона)

**Нейтрон с явными атрибутами:**

```
(нейтрон Удвоить(  
  число Целое  
) Целое  
  @время 3  
  @память 2  
  @энергия 8  
  (  
    (Целое умножить число (Целое 2))
```

```

    (вернуть рез)
  )
)

```

**Составной протон с атрибутами:**

```

(протон Точка(
  x Целое
  y Целое
)
@время 25
@память 20
@энергия 50
)

```

**Атом как синтез:**

```

(атом Прямоугольник(
  левыйВерх Точка
  правыйНиз Точка

  нейтрон Площадь
  нейтрон Периметр
)
@время 60 (// Целое)
@память 32 (// Целое)
@энергия 80 (// Целое)
(
  (// инициализация)
)
)

```

### 3.5. Связывание и запрет

```

(протон Результат() @время 10 @память 8 @энергия 20 ((Целое 42)))
(уст x)
  (// связано с x)

(протон Результат() @время 10 @память 8 @энергия 20 ((Целое 42)))
()
  (// () – результат запрещён и недоступен)

(протон Результат() @время 10 @память 8 @энергия 20 ((Целое 42)))
  (// результат доступен, но не связан – временно)

```

### 3.6. Операционная семантика

(// Нотация { } и  $\rightarrow$  — метаязык формального описания, не синтаксис «Контекст»)

**Конфигурация вычисления:**

{ текущийКонтекст, доступныйРесурс, куча, стек }

где:

- текущийКонтекст — контекст, исполняемый в данный момент
- доступныйРесурс — тройка (время, память, энергия) в виде значений Целое
- куча — отображение адресов на протоны, нейтроны, атомы
- стек — стек вызовов нейтронов

**Переход** обозначается стрелкой  $\rightarrow$  и описывает изменение состояния.

**Переход конструирования кварка** (внутри протона):

{ (протон имя() @время тп @память пп @энергия эп ((Кварк v))),  
(Тд, Пд, Эд), Н, С }  
(// где  $T_d \geq t_p + t_k$ ,  $P_d \geq p_p + p_k$ ,  $E_d \geq e_p + e_k$  — ресурс протона +  
кварка)  
 $\rightarrow$  { протон-значение, (Тд-тп-тк, Пд-пп-пк, Эд-эп-эк), Н, С }

**На языке «Контекст»** это соответствует:

(внешнийКонтекст  
@время Тд  
@память Пд  
@энергия Эд  
(

```

        (протон Имя() @время тп @память пп @энергия эп ((Кварк v)))
        (// успешно, расход (тп+тк, пп+пк, эп+эк))
    )
)

```

#### Переход вызова нейтрона:

```

{ (протон Нейтрон арг), (Тд, Пд, Эд), Н, С }
  (// Тд ≥ @время, Пд ≥ @память, Эд ≥ @энергия нейтрона)
→ { телоНейтрона, (Тд-@время, Пд-@память, Эд-@энергия), Н,
(точкаВозврата, С) }

```

Кризис — недостаток ресурса:

```

{ (протон Нейтрон арг), (Тд, Пд, Эд), Н, С }
  (// Тд < @время или Пд < @память или Эд < @энергия)
→ { кризис, проверкаРазрешенийСнижения, илиОстанов }

```

### 3.7. Пример программы

```

(нейтрон Главная())Целое
  @время 100
  @память 80
  @энергия 200
  (
    (протон Десять() @время 10
      @память 8
      @энергия 20 ((Целое 10)))(уст а)
    (протон Двадцать() @время 10
      @память 8
      @энергия 20 ((Целое 20)))(уст б)
    (а Сложить б)(уст сумма)
    (сумма)()
    (вернуть сумма)
  )
)

```

## Библиографический список

1. Eden, A. H. Three Paradigms of Computer Science / A. H. Eden // *Minds and Machines*. — 2007. — Vol. 17, № 2. — P. 135–167. — DOI: 10.1007/s11023-007-9060-8. (Перевод: Эден Э. Х. Три парадигмы информатики // *Вопросы философии*. — 2008. — № 7. — С. 141–152).
2. Andrae, A. S. G. On Global Electricity Usage of Communication Technology: Trends to 2030 / A. S. G. Andrae, T. Edler // *Challenges for Science: Engineering and Education*. — 2015. — Vol. 6, № 1. — P. 117–157. — DOI: 10.3390/challe6010117. (Проблема энергопотребления ЦОД).
3. Stefik, A. An Empirical Investigation into Programming Language Syntax / A. Stefik, S. Hanenberg // *ACM Transactions on Programming Languages and Systems (TOPLAS)*. — 2014. — Vol. 36, № 2. — P. 1–40. — DOI: 10.1145/2657905. (Когнитивная нагрузка и синтаксис).
4. Turner, R. Computational Artefacts: Towards a Philosophy of Computer Science / R. Turner. — Berlin: Springer, 2014. — 188 p. — DOI: 10.1007/978-3-662-44158-2. (Онтология вычислений, критика платонизма в КС).
5. Turing, A. M. On Computable Numbers, with an Application to the Entscheidungsproblem / A. M. Turing // *Proceedings of the London Mathematical Society*. — 1936. — Vol. s2-42, № 1. — P. 230–265. — DOI: 10.1112/plms/s2-42.1.230.
6. Asperti, A. The Optimal Implementation of Functional Programming Languages / A. Asperti. — Cambridge: Cambridge University Press, 1998. — 388 p. (Анализ стоимости  $\beta$ -редукции в лямбда-исчислении).
7. Hoare, C. A. R. An Axiomatic Basis for Computer Programming / C. A. R. Hoare // *Communications of the ACM*. — 1969. — Vol. 12, № 10. — P. 576–580. — DOI: 10.1145/363235.363259. (Отсутствие времени, как физического параметра в логике Хоара)
8. Feenberg, A. Transforming Technology: A Critical Theory Revisited / A. Feenberg. — Oxford: Oxford University Press, 2002. — 240 p. (Отчуждение в технологическом обществе, критика инструментальной рациональности).
9. Ленин, В. И. Материализм и эмпириокритицизм. Критические заметки об одной реакционной философии / В. И. Ленин // *Полное собрание сочинений*. — Т. 18. — М.: Изд-во полит. лит-ры, 1961. — С. 7–384. (Теория отражения, диалектический материализм).
10. Дополнительно: Степин, В. С. Философия науки и техники / В. С. Степин. — М.: Академический проект, 2003. — 800 с. — (Философия. Энциклопедия).

11. Landauer, R. Irreversibility and Heat Generation in the Computing Process / R. Landauer // IBM Journal of Research and Development. — 1961. — Vol. 5, № 3. — P. 183–191. — DOI: 10.1147/rd.53.0183. (Термодинамика вычислений).

Landauer, R. The Physical Nature of Information / R. Landauer // Physics Letters A. — 1996. — Vol. 217, № 4–5. — P. 188–193. — DOI: 10.1016/0375-9601(96)00453-7. (Принцип Ландауэра, обобщение).

12. Bennett, C. H. Logical Reversibility of Computation / C. H. Bennett // IBM Journal of Research and Development. — 1973. — Vol. 17, № 6. — P. 525–532. — DOI: 10.1147/rd.176.0525.

Bennett, C. H. The Thermodynamics of Computation—A Review / C. H. Bennett // International Journal of Theoretical Physics. — 1982. — Vol. 21, № 12. — P. 905–940. — DOI: 10.1007/BF02084158.

13. Lloyd, S. Ultimate Physical Limits to Computation / S. Lloyd // Nature. — 2000. — Vol. 406, № 6799. — P. 1047–1054. — DOI: 10.1038/35023282. (Пределы вычислений, энтропийные барьеры).

14. Уемов, А. И. Системный подход и общая теория систем / А. И. Уемов. — М.: Мысль, 1978. — 272 с. (Методология системного анализа, формализация противоречий).

15. Mumford, L. The Myth of the Machine: Technics and Human Development / L. Mumford. — New York: Harcourt, Brace & World, 1967. — 342 p. (Концепция «мегамашины» и «касты технологических жрецов»).

Альтернатива: Эллюль, Ж. Технологическое общество / Ж. Эллюль; пер. с фр. — М.: Прогресс, 1987. — 528 с.

16. Margolus, N. The Maximum Speed of Dynamical Evolution / N. Margolus, L. B. Levitin // Physica D: Nonlinear Phenomena. — 1998. — Vol. 120, № 1–2. — P. 188–195. — DOI: 10.1016/S0167-2789(98)00054-2. (Скорость вычислений и энергия).

17. Girard, J.-Y. Linear Logic / J.-Y. Girard // Theoretical Computer Science. — 1987. — Vol. 50, № 1. — P. 1–101. — DOI: 10.1016/0304-3975(87)90045-4. (Ресурсная интерпретация логики).

18. Matsakis, N. D. The Rust Language / N. D. Matsakis, F. S. Klock // ACM SIGAda Ada Letters. — 2014. — Vol. 34, № 3. — P. 103–104. — DOI: 10.1145/2692956.2663188. (Система владения как форма ресурсного контроля).

19. Tov, J. A. Practical Affine Types / J. A. Tov, R. Pucella // Proceedings of the 38th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL). — 2011. — P. 357–368. — DOI: 10.1145/1926385.1926429. (LinearML и практические линейные типы).

20. Forrester, J. W. *Industrial Dynamics* / J. W. Forrester. — Cambridge, MA: MIT Press, 1961. — 464 p. (Системная динамика как метод анализа противоречий).

21. Хайдеггер, М. Вопрос о технике / М. Хайдеггер // *Время и бытие: Статьи и выступления*. — М.: Республика, 1993. — С. 227–252. (Онтологический статус техники, «Gestell»).

22. Фуко, М. Дисциплинировать и наказывать. Рождение тюрьмы / М. Фуко; пер. с фр. В. П. Визгина. — М.: Ad Marginem, 2000. — 416 с. (Технологии власти, «паноптикум» как метафора систем контроля).

24. Feenberg, A. *Technosystem: The Social Life of Technological Reason* / A. Feenberg. — Cambridge, MA: Harvard University Press, 2017. — 240 p. (Современная критика технологической рациональности, социальная онтология техники).