
У РАЗЛИЧИЯ ДОЛЖЕН БЫТЬ ПРЕДОК

Теорема предкового разделения и первый момент определимости

Кирилл Шебетов · beforeword / research

Phase 2 · technical version 2.0.0 · 28 августа 2026

Содержание

Аннотация

0. Нулевая уступка словам
1. Что опровергнуто полностью
2. Четыре отношения нельзя склеивать
3. Фиксированный base и типизированная ancestry
4. AST-1: теорема предкового разделения
5. Что нового установила Phase 2
6. Неретроактивность
7. Provenance и разрез
8. FDT-2: первый момент определимости
9. Почему попарной проверки недостаточно
10. Уточняющаяся фильтрация и первый stage
11. Рандомизация не создаёт различие
12. Application binding как следствие
13. Ракета: максимальная уступка без подмены
14. «Зачем?» и goal provenance
15. Сто, двести, триста
16. Язык раньше математического чтения
17. Сильнейшие возражения
18. Фальсификаторы
19. Исполняемая область
20. Самоприменение
21. Предшественники и статус новизны
22. Граница притязаний
23. Итог

Статус источников

AST-1 / FDT-2: точная спецификация

1. Статус и запрет семантического сокращения
2. Данные AST-1
3. Лемма least provenance
4. Теорема AST-1
5. No-export следствие
6. Неретроактивность
7. Origin tags не заменимы untagged union
8. Теорема provenance cut
9. Данные FDT-2
10. Теорема FDT-2
11. Refinement и первый selectable stage

12. Randomized bound
13. Связь AST-1 и FDT-2
14. Точные положительные контроли
15. Фальсификация предпосылок
16. Максимальная точная формулировка
17. Источники и приоритет

Типизированное происхождение: предки, необратимость и разрезы

1. Что здесь установлено
2. Конечный immutable typed DAG
3. Origin-tagged introductions
4. IADC: ancestor-union как фундамент
5. Теорема не-ретроактивности
6. Теорема типизированного vertex cut
7. Положительные и отрицательные контроли
8. Композиция с AST-1: различие обязано иметь предка
9. Жёсткая граница вывода

Максимальная уступка: ракета, событие и вопрос «зачем»

1. Уступка принимается целиком
2. Четыре разных отношения
3. Что именно даёт успешный полёт
4. Первый момент определмости
5. Горизонт 100 / 200 / 300 лет
6. Четыре target вместо одного слова «цель»
7. Перестановка binding как частный случай
8. Transcript- и sensor-controls
9. Что остаётся от вопроса «зачем?»

04. КРАСНАЯ КОМАНДА: ГДЕ ИМЕННО МОЖНО СЛОМАТЬ PHASE 2

Назначение этого файла

1. Четыре отношения, которые нельзя склеивать
2. Точная мишень AST-1 и атаки на нее
3. Точная мишень FDT и атаки на нее
4. Provenance и cut: что именно они доказывают
5. Самые опасные семантические расширения
6. Контроли, без которых результат недостоверен
7. Самоприменение: этот документ не исключение
8. Финальный реестр: claim / status / falsifier

Аннотация

Исследование закрывает точное притязание, согласно которому записанное математическое преобразование способно самостоятельно доставить отношение к названному, событие, применение или цель, не получив различающей записи. Ни один глиф, токен, формула, слово событие, слово цель и слово предок не принимаются за то, что ими названо. Все они сначала являются элементами объявленного record-контракта.

Вводится конечный типизированный граф происхождения. Базовая проекция ρ фиксирует полный общий core, включая топологию графа, правила, идентификатор заключения и всё, что evaluator вправе читать помимо типизированной ancestry. Для узла заключения v проекция η_v собирает записи, введённые в его рефлексивной ancestry, сохраняя тип, atom-id и узел происхождения. Если заключение вычисляется как

$$C_v = F(\rho, \eta_v)$$

и претендует на точное совпадение с независимо заданной координатой Q , то

$$\ker \rho \cap \ker \eta_v \subseteq \ker Q.$$

Следовательно, всякая пара с одинаковым base-core и различным Q обязана быть разделена origin-tagged записью, введённой в ancestry заключения. Более поздний узел не может быть первым источником уже принятого различия: он не принадлежит ancestry прежнего заключения. Это **теорема предкового разделения AST-1**.

Для последовательности расширяющихся views доказывается relation-valued **теорема первого момента определимости FDT-2**. На стадии t существует zero-error selector тогда и только тогда, когда в каждом волокне view непусто пересечение множеств допустимых ответов. Если views последовательно уточняются, определимость монотонна и, когда она вообще возникает, имеет первый stage. Для m попарно непересекающихся множеств допустимых ответов в одном волокне худшая вероятность успеха любого randomized selector не превосходит $1/m$.

Результат не является «новым углом». В объявленной области он полностью опровергает автономный экспорт record $\rightarrow$ target: вычисление не создаёт target-различия из одинакового доступного происхождения. При этом не утверждается, что все формальные теоремы неверны. Внутренняя выводимость может быть безошибочной по собственному контракту; из неё не следует ненаписанный referent, occurrence, causal link или purpose.

Ключевые слова: граница записи; ancestry; origin tag; provenance; автономный экспорт; первое различающее введение; фильтрация views; zero-error selector; цель; событие; причинность; self-application.

0. Нулевая уступка словам

Эта работа не начинается с утверждения, что знак «обозначает объект». Она начинается с точной записи и объявленного способа обращения с ней.

Строки

```
количество
ракета
Марс
событие
цель
смысл
```

не предъявляют ничего, кроме самих строк. Поэтому никакое эмпирическое предложение, построенное из названий, не используется как посылка. Слово `количество` не становится количеством; строка `ракета полетела` не становится полётом. Добавление кавычек, формулы, фотографии, таблицы, `telemetry-label` или подписи `true` не отменяет необходимость отдельного контракта чтения и отдельного источника различия.

Даже выражения `record`, `target`, `ancestor` и `cause` здесь не обладают семантической привилегией. Они являются техническими метками ролей. Теорема использует только равенство точных записей, принадлежность узлов фиксированному графу и функциональность объявленной процедуры.

Это удерживается без исключений — в примерах, возражениях, выводах и самоприменении.

1. Что опровергнуто полностью

Слово «математика» слишком широко, чтобы его отрицание без области было теоремой. Поэтому объект опровержения задаётся точнее и сильнее риторической формулы.

Пусть K — объявленный класс допустимых индексов, $\rho(k)$ — весь фиксированный base-core, $\eta_v(k)$ — вся разрешённая origin-tagged ancestry заключения v , а $Q(k)$ — независимо зафиксированная target-запись. **Автономный предковый экспорт** имеет место, если существует evaluator F , который без иного входа восстанавливает target на всём K :

$$\forall k \in K: Q(k) = F(\rho(k), \eta_v(k)).$$

Если найдены k_0, k_1 такие, что

$$\rho(k_0) = \rho(k_1), \eta_v(k_0) = \eta_v(k_1), Q(k_0) \neq Q(k_1),$$

то притязание опровергнуто для **всех** функций F :

$$\forall F \exists i \in \{0, 1\} : F(\rho(k_i), \eta_v(k_i)) \neq Q(k_i).$$

Слово «полностью» относится к универсальному квантору по evaluator. Не остаётся более длинного расчёта, более сильного proof search, более богатой внутренней модели или более выразительной строки, которая могла бы различить два одинаковых доступных входа, не прочитав различающий канал.

Это и есть окончательная отрицательная граница. Она не зависит от мнения о математике. Она следует из невозможности функции вернуть два разных обязательных результата на одном и том же аргументе.

2. Четыре отношения нельзя склеивать

В рукописи жёстко разделены четыре отношения:

Обозначение	Точная роль в контракте	Чего оно не даёт само
$Contains_R(r, a)$	exact-record r содержит atom-code a по процедуре R	названного atom-code объекта
$Entails_C(\Gamma, \varphi)$	checker исчисления C принимает trace от Γ к formula-code φ	occurrence, application или purpose
$Occurs_P(e)$	независимый протокол P присвоил event-code e объявленный статус	референт вне P , причинность или цель
$Causes_M(x, e)$	независимая model/procedure M приняла causal-link-code (x, e)	безусловную причинность, occurrence или endorsement

Без отдельно записанного bridge ни одна стрелка не наследуется:

$$Contains \Rightarrow \not\Rightarrow Entails \Rightarrow \not\Rightarrow Occurs \Rightarrow \not\Rightarrow Causes.$$

Обратные стрелки также не даны. Даже когда bridge введён, он является новым typed atom с собственным происхождением и областью. Строка EVENT_RECORD сообщает тип записи; она не сообщает, что событие произошло. Строка GOAL сообщает тип записи; она не оправдывает цель. Строка CAUSES в payload не создаёт typed causal dependency.

3. Фиксированный base и типизированная ancestry

Пусть $D = (V, E)$ — конечный ориентированный ациклический граф с функцией рождения

$$b: V \rightarrow \{0, \dots, |V| - 1\},$$

такой, что для каждого ребра $u \rightarrow w$:

$$b(u) < b(w).$$

Узлы имеют типы введения из множества

$$T = \{formal, binding, event, goal\}.$$

Для индекса k , узла u и типа τ задано конечное множество локально введённых atom-id:

$$Intro_{\tau}(u, k).$$

Payload узла не читается как typed introduction. Совпадение строки payload со словом goal, event, binding или с идентификатором atom не создаёт зависимость. Типизированное введение существует только в защищённом поле контракта.

Пусть $Anc^*(v)$ — рефлексивная ancestry узла v : сам v и все узлы, из которых идёт путь к v . Origin-tagged проекция ancestry определяется как

$$\eta_v(k) = canon\left(\left(u, \tau, a\right) : u \in Anc^*(v), \tau \in T, a \in Intro_{\tau}(u, k)\right).$$

Тег u принципиален. Один atom-id, введённый в разных узлах, не является одной записью происхождения. Без origin tag можно проверить присутствие atom, но нельзя установить, где различие вошло в ancestry.

Проекция ρ фиксирует:

- топологию D , birth ranks и parent edges;
- типовую политику и правила распространения provenance;
- идентификатор заключения v ;
- evaluator contract и всё иное, что evaluator вправе читать;
- точные поля, объявленные общим base-core.

Если топология, правило, root-id или скрытое состояние меняется между индексами, это изменение обязано находиться в ρ . Иначе слово полный было бы ложной меткой read-set.

4. AST-1: теорема предкового разделения

Пусть evaluator заключения имеет вид

$$C_v(k) = F(\rho(k), \eta_v(k)).$$

Пусть также заявлена точность относительно независимо заданной координаты $Q: K \rightarrow Y$:

$$\forall k \in K: C_v(k) = Q(k).$$

Тогда необходимо

$$\ker \rho \cap \ker \eta_v \subseteq \ker Q.$$

Эквивалентная конфликтная форма:

$$\rho(k_0) = \rho(k_1) \wedge Q(k_0) \neq Q(k_1) \Rightarrow \eta_v(k_0) \neq \eta_v(k_1).$$

Так как η_v является канонической совокупностью origin-tagged introductions, существует тройка

$$(u, \tau, a) \in \eta_v(k_0) \Delta \eta_v(k_1),$$

где

$$u \in \text{Anc}^*(v), b(u) \leq b(v).$$

Иными словами: если корректное заключение различило target при одинаковом base-core, различающая typed запись была введена в его ancestry. Нельзя приписать это различие одной только форме заключения, слову внутри него или узлу, родившемуся после него.

Доказательство

Возьмём $(k, k') \in \ker \rho \cap \ker \eta_v$. Тогда

$$\rho(k) = \rho(k'), \eta_v(k) = \eta_v(k').$$

Функциональность F даёт

$$F(\rho(k), \eta_v(k)) = F(\rho(k'), \eta_v(k')).$$

По условиям $C_v = F(\rho, \eta_v)$ и $C_v = Q$ получаем $Q(k) = Q(k')$. Значит, $(k, k') \in \ker Q$, и включение ядер доказано.

Контрапозиция даёт $\eta_v(k_0) \neq \eta_v(k_1)$ для любого base-core конфликта. Неравенство двух конечных канонических совокупностей означает непустую симметрическую разность, поэтому существует различающая тройка (u, τ, a) . По определению η_v , её origin u принадлежит $\text{Anc}^*(v)$. Строгий порядок родителей даёт $b(u) \leq b(v)$. ▢

Доказательство не использует содержание atom-id. Оно одинаково для строк theorem, measurement, event, rocket, goal, meaning и для любых других точных кодов.

5. Что нового установила Phase 2

Phase 1 доказала общую проекционную границу: target-различие требует added channel. Phase 2 не выдаёт частные примеры за новые независимые теоремы. Она локализует канал в структуре происхождения.

Новый шаг таков:

Недостаточно сказать, что различие «было добавлено». Если заключение объявлено функцией фиксированного base и типизированной ancestry, origin-tagged запись, которая разделила конфликт, обязана иметь конкретный origin среди предков заключения.

Отсюда следуют, но не образуют самостоятельных ядер:

- **binding-следствие:** application binding, различивший target, имеет origin в ancestry принятого application verdict;
- **purpose-следствие:** goal-code, различивший допустимые purpose-ответы, имеет origin в ancestry purpose verdict;
- **prefix-следствие:** первый различающий ответ интерактивной записи входит в первый prefix/view, где волокно было разделено;
- **event-следствие:** event-code не может войти в provenance более раннего formal verdict через более поздний узел;
- **payload-spoof следствие:** строка нужного типа в незащищённом payload не заменяет typed introduction.

Binding, purpose и prefix — применения AST-1/FDT-2, а не три новых основания. Центр Phase 2 один: различие локализуется в своём происхождении.

6. Неретроактивность

Для принятого узла v его ancestry содержит только узлы с birth rank не выше $b(v)$. Если позднее создан w с

$$b(w) > b(v),$$

то $w \notin \text{Anc}^*(v)$. Следовательно, introductions узла w не входят в η_v .

Поздняя запись может стать предком нового узла v' . Она может изменить новый verdict, новый view и новый target-status. Она не становится источником прежнего verdict задним числом.

Это не утверждение, что позднее нельзя переписать файл. Переписанный файл — новый record. Утверждение относится к зафиксированному узлу, его committed prefix и объявленной ancestry.

Короткая форма:

Поздняя строка может обосновывать позднее заключение.

Она не была предком раннего заключения.

7. Provenance и разрез

Для каждого типа τ least provenance в узле v имеет вид

$$Prov_{\tau}(v, k) = \bigcup_{u \in Anc^*(v)} Intro_{\tau}(u, k).$$

Origin-tagged версия не уничтожает место введения:

$$OProv_{\tau}(v, k) = \{(u, a) : u \in Anc^*(v), a \in Intro_{\tau}(u, k)\}.$$

Если atom введён в нескольких источниках, он достигает root по нескольким путям. Гипотетическое удаление набора узлов X , не содержащего защищённый root, устраняет atom из root ровно тогда, когда X пересекает каждый путь $introducer \rightarrow root$. Минимальный такой X является provenance cut certificate.

Разрез не удостоверяет названное atom. Он отвечает только на структурный вопрос: какие защищённые узлы несут все объявленные пути данной typed записи к root.

8. FDT-2: первый момент определимости

Функциональная target-координата требует одного точного ответа для каждого индекса. Практический контракт может допускать несколько ответов. Поэтому вводится отношение допустимости.

Пусть:

- K — конечное непустое множество индексов;
- Y — конечное множество answer-codes;
- $A(k) \subseteq Y$ — непустое множество допустимых answer-codes для k ;
- $V_t : K \rightarrow R_t$ — view на стадии t .

Zero-error selector на стадии t — функция $s_t : R_t \rightarrow Y$, для которой

$$\forall k \in K : s_t(V_t(k)) \in A(k).$$

Для каждого view-value r обозначим волокно

$$K_{t,r} = \{k \in K : V_t(k) = r\}.$$

Теорема FDT-2

Zero-error selector существует тогда и только тогда, когда

$$\forall r \in V_t(K): \bigcap_{k \in K_{t,r}} A(k) \neq \emptyset.$$

Необходимо общее пересечение **всего** волокна, а не только каждой пары.

Доказательство

Если selector s_t существует, то для фиксированного r один и тот же code $s_t(r)$ должен принадлежать $A(k)$ для каждого $k \in K_{t,r}$. Поэтому он принадлежит их пересечению.

Обратно, если каждое пересечение непусто, для каждого конечного волокна выбирается один code из пересечения и назначается значением $s_t(r)$. Тогда один выбранный code допустим для каждого индекса волокна. $\square$

9. Почему попарной проверки недостаточно

Рассмотрим одно волокно из трёх индексов:

$$A(k_0) = \{a, b\}, A(k_1) = \{b, c\}, A(k_2) = \{a, c\}.$$

Все попарные пересечения непусты:

$$A(k_0) \cap A(k_1) = \{b\},$$

$$A(k_0) \cap A(k_2) = \{a\},$$

$$A(k_1) \cap A(k_2) = \{c\}.$$

Но

$$A(k_0) \cap A(k_1) \cap A(k_2) = \emptyset.$$

Один selector видит один view-value и обязан вернуть один answer-code. Ни один code не допустим сразу для трёх индексов. Поэтому pairwise audit недостаточен; точный критерий FDT-2 контролирует полное волокно.

10. Уточняющаяся фильтрация и первый stage

Пусть stages упорядочены $0, 1, \dots, n$, а каждый следующий view уточняет предыдущий:

$$\ker V_{t+1} \subseteq \ker V_t.$$

То есть равенство на более позднем view влечёт равенство на более раннем; позднее волокно является подмножеством раннего. Если условие FDT-2 выполнено на стадии t , оно выполнено на каждой стадии $t' > t$: пересечение по меньшему волокну не может потерять общий answer-code, уже общий для большего волокна.

Если множество selectable stages непусто, определён первый момент:

$$\tau^* = \min \left(t : \forall r \in V_t(K), \bigcap_{k \in K_{t,r}} A(k) \neq \emptyset \right).$$

Тогда:

$$t < \tau^* \Rightarrow \text{zero-error selector не существует,}$$

$$t \geq \tau^* \Rightarrow \text{zero-error selector существует.}$$

Если множество пусто, first stage не существует **на объявленной фильтрации**. Нельзя заменить этот статус фразой «никогда». Будущий канал может задать новый stage; он будет новым входом, а не открытием скрытого ответа внутри старого view.

Stage-id является индексом записи. Он не считается физическим временем. Строки 100, 200, 300 не становятся годами без reading contract.

11. Рандомизация не создаёт различие

Пусть в одном view-волокне находятся $k_1, \dots, k_m$, а множества $A(k_i)$ попарно не пересекаются. Randomized selector на общем view-value r задаёт одно распределение P_r на Y . Вероятность допустимого ответа для индекса k_i :

$$p_i = P_r(A(k_i)).$$

Из попарной непересекаемости следует

$$\sum_{i=1}^m p_i \leq 1.$$

Значит,

$$\min_{1 \leq i \leq m} p_i \leq \frac{1}{m}.$$

Random seed не является ответом из target только потому, что он назван random. Если его распределение одинаково на одном волокне, оно не разделяет индексы. Если seed, oracle

response или branch choice зависит от индекса, эта зависимость является новым каналом и должна войти в view/provenance.

Для zero-error randomized selector требуется

$$\text{supp } P_r \subseteq \bigcap_{k \in K_{t,r}} A(k).$$

Поэтому нулевой риск возможен ровно при том же непустом пересечении, что и в детерминированной FDT-2.

12. Application binding как следствие

Пусть два индекса имеют одинаковые formal records, proof traces и evaluator contract, но разные множества допустимых application answer-codes. Пока view не содержит различающий binding channel, оба индекса находятся в одном волокне.

Если пересечение допустимых answer-codes пусто, FDT-2 запрещает zero-error application selector. Если поздний stage добавляет typed binding и разделяет волокно, первый selectable stage совпадает не с «моментом, когда формула поняла применение», а с первым view, где различающая binding-запись стала доступна.

AST-1 добавляет происхождение: если application verdict корректно различает target, origin-tagged binding atom должен находиться в ancestry этого verdict.

Запись полного binding может содержать formal-result id, target-code, symbol-map, use-code, scope-code и fit-criterion-code. Структурная полнота этих полей не превращает target-code в референт и не удостоверяет fit. Она только делает источник утверждаемого различия проверяемым.

13. Ракета: максимальная уступка без подмены

Можно расширить контракт записями

$\ulcorner$ «расчёт принят» $\urcorner$, $\ulcorner$ «ракета построена» $\urcorner$, $\ulcorner$ «ракета полетела» $\urcorner$, $\ulcorner$ «траектория совпала» $\urcorner$.

Работа не отвергает эти строки и не выдаёт их за предъявленные события. Она проверяет происхождение статусов, которые им присваивает объявленный протокол.

Различаются:

Код	Типизированная роль
m	formal calculation record
b	application-binding record

Код	Типизированная роль
<i>e</i>	event-code, введённый event node
<i>c</i>	causal-link-code, принятый отдельным contract
<i>g</i>	goal-code

Ни одно равенство $m=b=e=c=g$ не следует из совместного присутствия кодов. Ни одна импликация между Entails, Occurs и Causes не следует из одинакового написания payload.

Если event-code введён после formal verdict, он не входит в ancestry прежнего verdict. Если новый end-to-end verdict имеет formal, binding и event nodes среди предков, именно расширенная ancestry является его входом. Успех нового протокола не превращает поздний event-code в содержимое ранней формулы.

Формула могла участвовать в цепи записей.

Цепь не находилась в формуле до введения её узлов.

14. «Зачем?» и goal provenance

Purpose не совпадает с выводимостью, применением, occurrence или causal-link status:

```

FORMAL - CHECKED
  != APPLICATION-BINDING-RECORDED
  != EVENT-CODE-ACCEPTED
  != CAUSAL-LINK-CODE-ACCEPTED
  != GOAL-CODE-RECORDED
  != PURPOSE-JUSTIFIED

```

Если два индекса имеют одинаковый view до goal channel и непересекающиеся допустимые purpose answer-sets, zero-error selector невозможен. Когда различающий goal-code появляется в новом view, определимость может впервые возникнуть. AST-1 требует, чтобы этот goal-code имел origin в ancestry purpose verdict.

Отсутствие goal-code не доказывает строку смысла нет. Оно также не доказывает строку смысл есть. Обе являются answer-codes. Жёсткий результат сильнее выбора стороны:

Ни положительный, ни отрицательный purpose verdict не экспортируется автономно из одинакового formal/application core.

Вычисление максимума для записанной utility-функции не выбирает, зачем принять эту функцию. Если utility-code уже введён, вычисление условно относительно него. Если endorsement требуется как новый target, его различие должно иметь новый origin.

15. Сто, двести, триста

Последовательности 100, 200, 300 являются token-codes. По reading contract они могут быть параметрами horizon-field и войти в процедуры

$$T(100), T(200), T(300).$$

Увеличение token не добавляет goal atom. Если horizon-node u_h не имеет защищённого goal-introduction, то он не является goal-origin:

$$Intro_{goal}(u_h, k) = \emptyset \Rightarrow \nexists a: (u_h, goal, a) \in \eta_v(k).$$

Если все declared stages до горизонта имеют конфликтное purpose-волокно, точный статус — NOT-SELECTABLE-ON-DECLARED-STAGES. Он не расширяется до утверждения обо всём будущем. Если через новый stage добавлен goal channel, это новый предок нового заключения.

Время не производит цель из записи. Новая запись может появиться во времени; тогда именно она меняет view.

16. Язык раньше математического чтения

Структурный порядок исследования таков:

exact carrier < reading contract < mathematical classification < formal trace < binding/event/goal records.

Слова линия, число, символ и значение сами являются классификациями. На нулевом уровне фиксируется carrier без утверждения, что он уже число или значение. Только объявленная процедура вводит чтение READ-AS-MATHEMATICAL.

Поэтому математическое чтение не может быть основанием собственного появления в данном record: reading contract логически предшествует результату чтения. Это порядок доступа и provenance, а не эмпирическая история цивилизации. Для центрального опровержения историческая гипотеза «математика возникла из языка» не нужна. Более сильный результат действует даже тогда, когда спор о происхождении оставлен без решения: ни один поздний mathematical label не переносит названное назад в carrier.

17. Сильнейшие возражения

17.1. «Референция уже внутри формулы»

Если точный reference-binding входит в защищённый input, он становится typed ancestor и расширяет η_v либо ρ . Тогда AST-1 не отрицает условный вывод. Она запрещает назвать этот input автономным результатом прежнего core.

Если внутри находится только строка `refers-to`, её присутствие устанавливает `Contains`, а не названный `referent`.

17.2. «Семантика является частью математики»

Внутренняя `model-record`, `interpretation-function` и `satisfaction-predicate` полностью допускаются как записи `base/ancestry`. На одинаковом `input` они одинаковы. `Intendedness`, `occurrence` или `application target`, различающийся на этом `input`, не выбирается ещё одним внутренним вычислением.

17.3. «Можно формализовать цель»

Можно ввести `goal-code` как посылку. Тогда он получает `origin`, входит в `ancestry` и может участвовать в выводе. Формализация показывает зависимость; она не уничтожает её. Вопрос принятия `goal-code` не становится выводом из доцелевого `core`.

17.4. «Позднее подтверждение делает ранний расчёт верным»

Поздний `record` может войти в новый `verdict`. Он не меняет `ancestry` уже зафиксированного узла. Термин подтверждение сам требует критерия и протокола; запись этого слова не пересобирает прошлый `provenance`.

17.5. «Причинное действие доказывает выход из записи»

Исполняемый `record` может быть входом `decoder/compiler/controller chain`. Каждое звено является новым узлом и имеет собственные `inputs`. Принятый `causal-link-code` относится к расширенной цепи, а не к автономной синтаксической выводимости. `Causes` не следует из `Entails` без `bridge`.

17.6. «Попарно всё совместимо»

Для `relation-valued target` требуется общее пересечение целого `view-fiber`. Тройной контроль в разделе 9 показывает, что попарная совместимость недостаточна.

17.7. «Случайность найдёт правильный ответ»

На одинаковом `view` распределение одно. Для m непересекающихся `acceptable sets` хотя бы один индекс имеет успех не выше $1/m$. Зависимая от индекса случайность является каналом.

17.8. «Через достаточно долгое время ответ появится»

Возможно появление нового различающего `channel-record`. Оно создаёт новый `stage`. Из длины прежнего `prefix` ответ не следует. Точный квантор относится к объявленной фильтрации; утверждение о бесконечном будущем без полного контракта не делается.

18. Фальсификаторы

AST-1 применена неверно, если:

1. evaluator фактически читает вход вне (ρ, η_v) ;
2. topology, rules, root-id или hidden state различаются, но не включены в ρ ;
3. заявлено $C_v = Q$, хотя equality не выполнено на всём K ;
4. origin tags потеряны и локализация источника заменена догадкой;
5. payload интерпретирован как typed introduction вопреки контракту;
6. committed prefix изменён после фиксации;
7. цикл или поздний parent нарушает объявленный birth order.

FDT-2 применена неверно, если:

8. view не является полным объявленным read-set процедуры;
9. acceptable sets определены после просмотра selector output;
10. stage, названный refinement, объединяет ранее различённые индексы;
11. проверены только пары, хотя волокно содержит больше двух индексов;
12. randomized seed зависит от скрытого index-channel;
13. answer-code выдан за referent или world-status.

Положительные случаи не скрываются. Если Q постоянно на совместных волокнах (ρ, η_v) , факторизация возможна. Если каждое acceptable intersection непусто уже на первом stage, selector существует сразу. Если новый channel разделяет конфликт, определимость может возникнуть на следующем stage. Теоремы не назначают отрицательный ответ заранее.

19. Исполняемая область

Конечный typed-provenance checker проверяет:

- строгий birth order и ациклическую parent dependency;
- допустимые typed introductions;
- least ancestor-union provenance;
- origin-tagged provenance у query root;
- committed canonical prefix;
- минимальные vertex-cut certificates для указанных atom-codes;
- инертность payload относительно typed dependencies.

Принятый структурный статус имеет форму VALID-TYPED-PROVENANCE-DAG; инвариант — PROVENANCE-INVARIANT-AUDIT-PASS. Семантический статус остаётся NOT-MACHINE-VERIFIED.

Finite filtration checker проверяет:

- exact declared views и их hashes;
- ordered refinement;
- полные acceptable intersections каждого fiber;
- первый selectable stage либо его отсутствие на declared filtration;
- higher-order triple conflict;
- randomized $1/m$ bound на disjoint control.

AST pair checker соединяет оба слоя на конечной паре. Он независимо пересчитывает effective base, включая structural graph signature, извлекает origin-tagged η из двух валидных provenance-графов и различает три исхода: VERIFIED-ANCESTRAL-SEPARATION-WITNESS, VERIFIED-AST-FACTORIZATION-CONFLICT и VERIFIED-BASE-CHANGED-CONTROL. Различие topology, rules или root не разрешено маскировать одинаковой пользовательской меткой base.

Статусы checker относятся к конечным JSON records. Они не удостоверяют exhaustiveness мира, referent answer-codes или исторический момент freeze. Общие AST-1 и FDT-2 являются текстовыми теоремами в объявленной метатеории; fixtures являются конечными instances.

20. Самоприменение

Эта рукопись сама является exact record. Слова теорема, доказано, предок, различие, математика И истина внутри неё не получают исключения.

Разделяются статусы:

```

точный текст AST-1/FDT-2
-> TEXTUAL-THEOREM-IN-DECLARED-METATHEORY

конечный checker verdict
-> VERIFIED-IN-DECLARED-ARTIFACT

публичный заголовок
-> UNBOUNDED-STANDALONE

SHA-256 файла
-> BYTE-IDENTITY-RECORDED

```

Название «У различия должен быть предок» читается только как AST-1 относительно фиксированных K, ρ, η_v, Q . Без этих параметров оно не является доказанным высказыванием обо всём существующем.

Если два допустимых extension имеют одинаковую рукопись и разные значения поля applies_to_all_mathematics, рукопись не выбирает это поле. Если поздний endorsement добавлен в новый файл, он является новым descendant record, а не предком замороженной версии.

21. Предшественники и статус новизны

Ни один внешний автор, школа, термин или смежная работа не является посылкой AST-1 или FDT-2. Доказательства построены из объявленных объектов и приведены полностью. Для их проверки не требуется доверие к исторической линии.

В этой фазе внешний collision/priority audit не используется как основание и не завершён. Поэтому статусы формулируются прямо:

```
PREDECESSORS-AS-PREMISES: NONE
HISTORICAL-PRIORITY: NOT-ASSESSED
GLOBAL-NOVELTY: NOT-ASSESSED
INTERNAL-PHASE-CONTRIBUTION: AST-1 + FDT-2
```

NOT-ASSESSED не означает первый, не первый или нового нет. Это отказ подменять самостоятельное доказательство библиографическим голосованием. Предшественник, найденный позднее, может изменить историческую маркировку; он не становится скрытой посылкой уже предъявленного доказательства.

22. Граница притязаний

Установлено

14. Если $Q = C_v = F(p, \eta_v)$, то $\ker p \cap \ker \eta_v \subseteq \ker Q$.
15. Каждый base-core target-conflict, корректно разделённый заключением, требует различающей origin-tagged introduction в ancestry заключения.
16. Поздний узел не является origin прежнего ancestry-различия.
17. Zero-error selector для relation-valued target существует ровно при непустом полном acceptable intersection каждого view-fiber.
18. В уточняющейся конечной фильтрации определимость монотонна и имеет первый stage, если возникает.
19. Для m disjoint acceptable sets в одном fiber худший randomized success не выше $1/m$.
20. Binding, purpose и transcript-prefix получают точные следствия из одного ядра, а не вводятся как риторические исключения.

Не установлено

- что все формальные теоремы неверны;
- что ни один знак никогда ни к чему не относится;
- что target-code предъявляет названное;
- что event-code является событием;
- что causal-link-code является причинностью;

- что goal-code оправдан;
- что применение, событие, причина, цель или смысл отсутствуют;
- что declared view исчерпывает любой возможный мир;
- что определимость не появится на неопisanном будущем stage;
- что историческая гипотеза происхождения математики доказана;
- что AST-1/FDT-2 имеют установленный мировой приоритет.

Эти ограничения не смягчают центральный результат. Они не дают слову заменить названное и не позволяют заголовку получить лишний квантор.

23. Итог

Формальная trace может быть корректной. Formula-code может быть получен из axioms-code. Program-code может пройти verifier-code. Это отношения внутри объявленной записи.

Но target-различие не возникает из одинакового base и одинаковой ancestry. Если заключение действительно различило, origin различия находится среди его предков. Если selector впервые стал безошибочным, первый различающий channel уже присутствует в его view. Если goal появился позднее, он принадлежит новому заключению. Если написано только слово *цель*, предъявлено только слово.

Опровергнуто автономное притязание:

Запись не экспортирует referent, occurrence, cause, application или purpose только силой дальнейшего преобразования той же записи.

Не опровергается условная формальная выводимость. Опровергается её незаконное продолжение за пределы собственного происхождения.

У различия должен быть предок.

У ненаписанного различия нет вычислительного пути в заключение.

Статус источников

Работа продолжает внутреннюю авторскую линию beforeword и предыдущие фазы данного проекта. Внешние сравнительные источники в посылки не включены. Исторический приоритет и глобальная новизна в Phase 2 не оценивались.

AST-1 / FDT-2: точная спецификация

1. Статус и запрет семантического сокращения

Этот файл задаёт две текстовые теоремы Phase 2:

- **AST-1** — теорему предкового разделения для функционального target;
- **FDT-2** — теорему первого момента определимости для relation-valued target.

Каждый объект спецификации является элементом объявленной метатеории. Слова `граф`, `предок`, `событие`, `цель`, `ответ`, `причина` и `математика` не предъявляют названного. В частности:

$$\text{Contains}_R(r, a), \text{Entails}_C(\Gamma, \phi), \text{Occurs}_P(e), \text{Causes}_M(x, e)$$

являются четырьмя разными отношениями. Ни одно из них не определяется через имя другого и ни одно не следует из другого без отдельного bridge-rule/input.

Общие теоремы имеют статус:

TEXTUAL - THEOREM - IN - DECLARED - METATHEORY

Конечные checker instances могут получить иной, более узкий статус:

VERIFIED - IN - DECLARED - ARTIFACT

2. Данные AST-1

2.1. Индексы и target

Пусть:

- $K \neq \emptyset$ — множество допустимых exact-record индексов;
- Y — множество target-codes;
- $Q: K \rightarrow Y$ — независимо заданная target-coordinate.

Независимость означает, что Q фиксируется как поле спецификации, а не определяется через output evaluator, отсутствие bridge, желаемый verdict или слово `external`.

Ядро равенства функции f обозначается

$$\ker f = \{ (k, k') \in K^2 : f(k) = f(k') \}.$$

2.2. Фиксированный типизированный DAG

Пусть $D = (V, E)$ — конечный DAG. Ориентация ребра $u \rightarrow w$ означает, что u является parent/dependency узла w . Задана инъективная birth-rank функция

$$b: V \rightarrow \{0, \dots, |V| - 1\}$$

с условием

$$u \rightarrow w \Rightarrow b(u) < b(w).$$

Для $v \in V$:

$$Anc(v) = \{u \in V : \text{существует непустой путь } u \leadsto v\},$$

$$Anc^*(v) = Anc(v) \cup \{v\}.$$

Слово «предок» далее означает только принадлежность $Anc^*(v)$.

2.3. Typed introductions

Пусть множество сортов

$$T = \{formal, binding, event, goal\}.$$

Для каждого $k \in K$, $u \in V$, $\tau \in T$ задано конечное множество atom-id

$$Intro_\tau(u, k).$$

Typed introduction задаётся только защищённым полем. Для произвольного payload $p(u, k)$ не вводится правило

$$\ulcorner a \urcorner \text{ встречается в } p(u, k) \Rightarrow a \in Intro_\tau(u, k).$$

Такого правила нет. Это исключает payload spoofing.

2.4. Base projection

Пусть

$$\rho: K \rightarrow R$$

— полная проекция fixed base. Она включает либо однозначно фиксирует:

- D , b , parent relation и node kinds;
- типовую политику и правило provenance closure;
- query/conclusion node v ;
- evaluator contract;
- любой другой input, который evaluator читает помимо ancestry projection.

Формально достаточно условия: если любой из этих компонентов различается для k, k' , то $\rho(k) \neq \rho(k')$. Скрыто меняющийся компонент нарушает предпосылку полного base.

2.5. Origin-tagged ancestry projection

Для фиксированного conclusion node v положим

$$\eta_v(k) = \text{canon} \left(\left((u, \tau, a) : u \in \text{Anc}^*(v), \tau \in T, a \in \text{Intro}_\tau(u, k) \right) \right).$$

canon является инъективным кодированием конечного множества triples. Поэтому

$$\eta_v(k) = \eta_v(k')$$

тогда и только тогда, когда origin-tagged множества совпадают точно.

Origin tag обязателен: пары (u, a) и (u', a) различны при $u \neq u'$, даже если atom-id одинаков.

2.6. Conclusion evaluator

Пусть

$$F : R \times E_v \rightarrow Y$$

— произвольная функция, а conclusion coordinate

$$C_v(k) = F(\rho(k), \eta_v(k)).$$

Притязание на корректный target export записывается отдельно:

$$\forall k \in K : C_v(k) = Q(k).$$

Без условия $C_v = Q$ evaluator может печатать любой code; сам output не становится правильным target.

3. Лемма least provenance

Для сорта τ определим рекурсивно:

$$\text{Prov}_\tau(v, k) = \text{Intro}_\tau(v, k) \cup \bigcup_{u \in \text{Parents}(v)} \text{Prov}_\tau(u, k).$$

Лемма AST-P

Для каждого v, k, τ :

$$\text{Prov}_\tau(v, k) = \bigcup_{u \in \text{Anc}^*(v)} \text{Intro}_\tau(u, k).$$

Доказательство

Индукция по $b(v)$. Если у v нет parents, обе части равны $\text{Intro}_\tau(v, k)$. Пусть равенство установлено для всех parents узла v . Подстановка индукционных предположений в рекурсивное определение

даёт union local introductions с introductions всех рефлексивных ancestors каждого parent. Это точно множество introductions на $Anc^*(v)$. $\square$

Лемма утверждает равенство записанных atom-id. Она не утверждает, что atom-id имеет referent.

4. Теорема AST-1

AST-1A. Ядерная форма

Если

$$C_v = F \circ (\rho, \eta_v) \text{ и } C_v = Q,$$

то

$$\ker \rho \cap \ker \eta_v \subseteq \ker Q.$$

Доказательство

Пусть $(k, k') \in \ker \rho \cap \ker \eta_v$. Тогда

$$(\rho(k), \eta_v(k)) = (\rho(k'), \eta_v(k')).$$

Функциональность F даёт

$$C_v(k) = F(\rho(k), \eta_v(k)) = F(\rho(k'), \eta_v(k')) = C_v(k').$$

Из $C_v = Q$ следует $Q(k) = Q(k')$. Значит, $(k, k') \in \ker Q$. $\square$

AST-1B. Конфликтная форма

Если

$$\rho(k_0) = \rho(k_1) \text{ и } Q(k_0) \neq Q(k_1),$$

то любой корректный evaluator вида $F(\rho, \eta_v)$ требует

$$\eta_v(k_0) \neq \eta_v(k_1).$$

Это контрапозиция AST-1A.

AST-1C. Origin witness

При условиях AST-1B существует

$$(u, \tau, a) \in \eta_v(k_0) \triangle \eta_v(k_1).$$

Для этого witness:

$$u \in Anc^*(v), b(u) \leq b(v).$$

Если различающий triple введён не в самом v , то $u \in Anc(v)$ и $b(u) < b(v)$.

Доказательство

Из AST-1B origin-tagged канонические множества различны. Инъективность canon даёт непустую симметрическую разность исходных множеств. Любой её элемент имеет вид (u, τ, a) с $u \in Anc^*(v)$ по определению η_v . Birth inequality следует из DAG order. $\square$

5. No-export следствие

Если существует пара k_0, k_1 с

$$\rho(k_0) = \rho(k_1), \eta_v(k_0) = \eta_v(k_1), Q(k_0) \neq Q(k_1),$$

то

$$\forall F: Q \neq F \circ (\rho, \eta_v).$$

Доказательство

Оба индекса подадут F один и тот же ordered pair. Поэтому output один. Он не равен двум различным значениям Q одновременно. $\square$

Длина вычисления, внутренняя семантика, proof trace, recursion, oracle-code и название результата в доказательстве не используются. Если oracle output различается, он является частью нового input.

6. Неретроактивность

Следствие AST-NR

Пусть w создан после conclusion node v :

$$b(w) > b(v).$$

Тогда

$$w \notin Anc^*(v)$$

и никакой triple (w, τ, a) не принадлежит $\eta_v(k)$.

Доказательство

Если бы $w \rightsquigarrow v$, строгий порядок вдоль пути дал бы $b(w) < b(v)$, что противоречит условию. $\square$

Поздний atom может войти в ancestry нового descendant conclusion. Он не становится первым источником различия старого conclusion.

7. Origin tags не заменимы untagged union

Пусть

$$\dot{\eta}_v(k) = \{(\tau, a) : \exists u(u, \tau, a) \in \eta_v(k)\}.$$

Возможны k_0, k_1 , для которых

$$\dot{\eta}_v(k_0) = \dot{\eta}_v(k_1)$$

при

$$\eta_v(k_0) \neq \eta_v(k_1),$$

если один и тот же atom-id введён разными nodes. Untagged union может сохранить membership, но уничтожить origin witness. Поэтому AST-1C формулируется для η_v , а не для $\dot{\eta}_v$.

8. Теорема provenance cut

Зафиксируем k, τ, a, v . Пусть множество introducers

$$S_{\tau,a}(v, k) = \{u \in \text{Anc}^*(v) : a \in \text{Intro}_\tau(u, k)\}.$$

Пусть $P_{\tau,a}(v, k)$ — множество всех направленных путей от каждого $u \in S_{\tau,a}(v, k)$ к v . Рассмотрим гипотетическое удаление узлов $X \subseteq V \setminus v$ и инцидентных рёбер, после которого least provenance пересчитывается на оставшемся графе.

Теорема AST-CUT

Если $v \notin S_{\tau,a}(v, k)$, то atom a отсутствует в пересчитанном $\text{Prov}_\tau(v, k)$ тогда и только тогда, когда

$$\forall P \in P_{\tau,a}(v, k) : X \cap P \neq \emptyset.$$

Доказательство

По лемме AST-P atom достигает v после удаления ровно тогда, когда остаётся хотя бы один introducer и хотя бы один неразорванный путь от него к v . Отсутствие такого пути эквивалентно тому, что X пересекает каждый исходный introducer-to-root path. $\square$

Если $v \in S_{\tau,a}(v, k)$ и root защищён от удаления, atom неустраним таким cut: он введён локально в root. Минимальный по мощности hitting set даёт min-cut certificate. Это структурный certificate, не semantic attestation.

9. Данные FDT-2

Пусть:

- K — конечное непустое множество record-indices;
- Y — конечное непустое множество answer-codes;
- $A \subseteq K \times Y$ — отношение допустимости;
- $A(k) = \{y \in Y : (k, y) \in A\} \neq \emptyset$;
- $V_t : K \rightarrow R_t$ — view на стадии $t \in \{0, \dots, n\}$.

Для $r \in V_t(K)$ волокно

$$K_{t,r} = V_t^{-1}(r).$$

Zero-error selector на stage t — функция

$$s_t : V_t(K) \rightarrow Y$$

такая, что

$$\forall k \in K : s_t(V_t(k)) \in A(k).$$

Вне $V_t(K)$ selector не обязан быть задан.

10. Теорема FDT-2

FDT-2A. Fiber-intersection criterion

Эквивалентны:

21. на stage t существует zero-error selector;

22. для каждого $r \in V_t(K)$:

$$\bigcap_{k \in K_{t,r}} A(k) \neq \emptyset.$$

Доказательство

1 $\rightarrow$ 2. Пусть s_t — zero-error selector. Для фиксированного r и каждого $k \in K_{t,r}$:

$$s_t(r) = s_t(V_t(k)) \in A(k).$$

Следовательно,

$$s_t(r) \in \bigcap_{k \in K_{t,r}} A(k).$$

2 -> 1. Так как K конечно, число непустых fibers конечно. Для каждого $r \in V_t(K)$ выберем

$$y_r \in \bigcap_{k \in K_{t,r}} A(k)$$

и положим $s_t(r) = y_r$. Тогда для каждого k :

$$s_t(V_t(k)) = y_{V_t(k)} \in A(k).$$

Значит, s_t zero-error. $\square$

FDT-2B. Higher-order obstruction

Условие

$$\forall k \neq k' \in K_{t,r} : A(k) \cap A(k') \neq \emptyset$$

недостаточно. Контроль

$$A(k_0) = \{a, b\}, A(k_1) = \{b, c\}, A(k_2) = \{a, c\}$$

имеет все непустые попарные пересечения и пустое тройное пересечение. Поэтому selector не существует.

11. Refinement и первый selectable stage

Фильтрация views удовлетворяет refinement condition, если

$$\ker V_{t+1} \subseteq \ker V_t \quad (0 \leq t < n).$$

Лемма FDT-M

Если stage t selectable, то каждый $t' > t$ selectable.

Доказательство

Каждое волокно $K_{t',r'}$ содержится в некотором волокне $K_{t,r}$. Общий answer-code большего волокна принадлежит acceptable sets всех элементов меньшего. По FDT-2A поздний stage selectable. $\square$

Определение первого stage

Положим

$$S = \{t \in \{0, \dots, n\} : \text{stage } t \text{ selectable}\}.$$

Если $S \neq \emptyset$, то

$$\tau^* = \min S.$$

По лемме FDT-M:

$$S = \{ \tau^*, \tau^* + 1, \dots, n \}.$$

Если $S = \emptyset$, первый stage не существует на данной конечной фильтрации. Это не утверждение о неопisanном продолжении.

12. Randomized bound

Randomized selector на view-value r задаётся распределением P_r на Y . Его success на $k \in K_{t,r}$:

$$p(k) = P_r(A(k)).$$

Теорема FDT-R

Если $k_1, \dots, k_m \in K_{t,r}$ и sets

$$A(k_1), \dots, A(k_m)$$

попарно не пересекаются, то

$$\min_{1 \leq i \leq m} p(k_i) \leq \frac{1}{m}.$$

Доказательство

Из попарной непересекаемости:

$$\sum_{i=1}^m P_r(A(k_i)) = P_r\left(\bigcup_{i=1}^m A(k_i)\right) \leq 1.$$

Если бы каждое $p(k_i) > 1/m$, сумма была бы больше единицы. Противоречие. $\square$

Zero-error randomized case

Randomized selector zero-error на fiber ровно тогда, когда существует распределение P_r с

$$\text{supp } P_r \subseteq \bigcap_{k \in K_{t,r}} A(k).$$

Такое распределение существует ровно при непустом пересечении. Случайность не ослабляет критерий FDT-2A для нулевой ошибки.

13. Связь AST-1 и FDT-2

AST-1 отвечает на вопрос:

Где в *ancestry* должно быть записано различие, если функциональный *conclusion* корректно различает *target*?

FDT-2 отвечает на вопрос:

На каком первом *view* уже существует единый допустимый *answer* для каждого *fiber*?

Если *stage views* построены из возрастающей *origin-tagged ancestry*, то переход от *neselectable stage* к *selectable stage* локализует добавленный *channel interval*. AST-1 затем локализует *typed origins* внутри *ancestry* нового *conclusion*.

Binding, *purpose* и *transcript-prefix* являются специализациями этой композиции:

- *binding atom* уточняет *application view*;
- *goal atom* уточняет *purpose view*;
- *transcript response* уточняет *prefix view*.

Они не являются независимыми от AST-1/FDT-2 теоретическими ядрами.

14. Точные положительные контроли

14.1. Factorization control

Если

$$\ker p \cap \ker \eta_v \subseteq \ker Q,$$

то существует экстенсиональная функция

$$g: (p, \eta_v)(K) \rightarrow Y$$

с $Q = g \circ (p, \eta_v)$. Она задаётся значением $Q(k)$ на любом представителе совместного волокна; включение ядер делает выбор независимым от представителя.

Для бесконечного K это существование функции на образе, а не утверждение её вычислимости.

14.2. Already-selectable control

Если на *stage 0* каждое *fiber intersection* непусто, то $\tau^* = 0$. FDT-2 не требует появления нового *channel* там, где один *acceptable answer* уже общий для каждого исходного *fiber*.

14.3. Never-selectable-on-declared-filtration control

Если хотя бы одно пустое *acceptable intersection* сохраняется на каждом $t \in \{0, \dots, n\}$, то $S = \emptyset$. Допустимый статус ограничен данной фильтрацией.

15. Фальсификация предпосылок

AST-1 не может использоваться для заявленного record, если:

- F читает незаявленный channel;
- меняющийся DAG/rule/root не включён в ρ ;
- Q выбран после output ради создания конфликта;
- равенство $C_v = Q$ не доказано в объявленной области;
- origin tag удалён, но заявлена локализация origin;
- payload принят за typed introduction без правила;
- birth order допускает late ancestor вопреки спецификации.

FDT-2 instance отвергается, если:

- $A(k) = \emptyset$ вопреки непустому relation contract;
- declared view скрывает фактически прочитанный input;
- refinement condition нарушено;
- selector получает index-dependent seed вне view;
- проверка заменяет полное fiber intersection набором pairwise intersections;
- answer-code выдан за referent, event или purpose.

16. Максимальная точная формулировка

Если корректное заключение является функцией фиксированного base и origin-tagged ancestry, то любое target-различие при одинаковом base обязано быть записано различием в ancestry; конкретный origin различающей typed записи является рефлексивным предком заключения. Для relation-valued target zero-error выбор впервые возможен ровно на первом уточнённом view, где полное пересечение acceptable answer-sets каждого fiber стало непустым. Ни имя, ни поздняя запись, ни случайность не создают отсутствующее различие в прежнем входе.

Точные строки self-audit

«Если полный объявленный вход заключения равен паре из базового core и origin-tagged ancestry, а заключение правильно воспроизводит независимо заданную target-координату на всём допустимом классе, то всякая target-различающая пара с одинаковым core обязана различаться в ancestry: различающий typed atom введён самим заключением или одним из его предков. При forward birth-rank более поздний узел не может быть первым источником этого различия.»

«Для конечного допустимого класса zero-error selector на стадии существует тогда и только тогда, когда на каждом волокне объявленного view пересечение множеств приемлемых ответов

непусто. При последовательном уточнении view определимость монотонна; если она когда-либо появляется, существует первый момент определмости.»

Публичная строка «У различия должен быть предок» означает эту формулу только относительно объявленных $K, D, b, v, \rho, \eta_v, Q, F$. Она не означает, что всякая формальная теорема неверна, и не превращает слова этой спецификации в названное ими.

17. Источники и приоритет

AST-1 и FDT-2 доказаны из определений данного файла. Внешние предшественники не являются посылками. Исторический приоритет и глобальная новизна не оценивались:

```
PREDECESSORS-AS-PREMISES: NONE
HISTORICAL-PRIORITY: NOT-ASSESSED
GLOBAL-NOVELTY: NOT-ASSESSED
```

Типизированное происхождение: предки, необратимость и разрезы

1. Что здесь установлено

Этот раздел не спрашивает, на что «похож» текст узла. Он фиксирует только защищённую структуру конечной записи: какой узел что **типизированно ввёл**, на какие более ранние узлы он ссылается и какие введённые достигают выбранного корня по записанным рёбрам.

Строка в `payload` не приобретает тип от собственного написания. Строка `EVENT_RECORD`, фраза «ракета полетела» и ключ `event` внутри произвольного `payload` не являются событием и не вводят даже типизированную запись события. Для введённого требуется отдельное защищённое поле соответствующего `kind`. То же относится к `PURPOSE`: это запись цели, а не наличие, истинность, оправданность или ценность цели.

Результаты раздела относятся к объявленному графу записей. Они не устанавливают физическое происшествие, физическую причинность, референцию, истинность или смысл.

2. Конечный `immutable typed DAG`

Пусть

$$G = (V, E)$$

— конечный ориентированный ациклический граф. Рёбро $(u, v) \in E$ означает только записанное отношение: u указан как `parent` узла v . Оно не означает физическую причину.

Каждый узел $v \in V$ имеет защищённые поля:

- уникальный `node_id`;
- `birth_rank` $b(v) \in \{0, \dots, |V| - 1\}$;
- `kind`;
- упорядоченный список `parents`;
- типизированные локальные введённые `introduces`;
- заявленное типизированное происхождение `declared_provenance`;
- произвольный `payload`, который не читается как источник типизированных зависимостей.

Для каждого ребра действует строгий порядок

$$(u, v) \in E \Rightarrow b(u) < b(v).$$

birth_rank — позиционный ранг внутри конечной записи. Это не физическое время, не длительность, не дата и не отношение «раньше в мире». Из $b(u) < b(v)$ следует только допустимое направление ссылки в этой записи.

Опубликованные узлы immutable: их kind, parents, introductions и payload не переписываются. Допустимое расширение append-only добавляет новые узлы с большими рангами и может направлять их ссылки только к уже существующим узлам. Оно не меняет старые рёбра.

Используются четыре защищённых сорта:

$$\Sigma = \{ \text{FORMAL}, \text{BINDING}, \text{EVENT}, \text{PURPOSE} \}.$$

В исполняемом формате им соответствуют ключи formal, binding, event, goal. Здесь event читается только как EVENT_RECORD, а goal — только как PURPOSE, то есть как запись цели.

Узел одного kind может локально вводить атомы только своего сорта. Узел VERDICT не вводит атомов ни одного из четырёх сортов. Совпадение байтов payload с идентификатором атома не меняет этого условия.

3. Origin-tagged introductions

Пусть

$$\text{Intro}_\sigma(v)$$

— конечное множество идентификаторов атомов сорта σ , локально введённых узлом v . Чтобы не потерять источник при объединении, используется origin-tagged форма:

$$\text{OIntro}(v) = \{ (\sigma, a, v) : a \in \text{Intro}_\sigma(v) \}.$$

Третий компонент — node_id источника. Поэтому два разных узла, вводящие один и тот же (σ, a) , дают два различных origin-tagged атома.

Обозначим рефлексивное ancestry выбранного узла:

$$\text{Anc}_G^*(v) = \{ u \in V : u = v \text{ или существует путь } u \rightsquigarrow v \}.$$

Рефлексивность нужна, потому что локальное введение самого v входит в его provenance.

4. IADC: ancestor-union как фундамент

Инвариант IADC

IADC — Introduction–Ancestry Dependency Closure. Для каждого узла v origin-tagged provenance определяется наименьшим монотонным замыканием локальных введений по parent-рёбрам:

$$OProv_G(v) = OIntro(v) \cup \bigcup_{p \in Parents(v)} OProv_G(p).$$

Эквивалентная ancestor-union форма:

$$OProv_G(v) = \bigcup_{u \in Anc_G^*(v)} OIntro(u).$$

Если удалить origin-теги, для каждого сорта получается обычное множество:

$$Prov_{G,\sigma}(v) = \bigcup_{u \in Anc_G^*(v)} Intro_\sigma(u).$$

Доказательство

Упорядочим узлы по birth_rank. У узла ранга 0 нет parents, поэтому рекурсивная и ancestor-union формы совпадают. Предположим совпадение для всех узлов меньшего ранга. Parents узла v имеют меньшие ранги, следовательно, к ним применимо предположение. Подстановка их ancestor-union форм в (IADC-rec) даёт объединение локального введения v и введений всех предков каждого parent. Это в точности множество введений всех рефлексивных предков v . $\square$

IADC — фундамент, а не центр новизны Phase 2. Union-no-deletion для bridge-зависимостей уже был частным случаем в Phase 0. Здесь фундамент поднят до четырёх защищённых сортов, origin-тегов и произвольного конечного DAG; новый шаг ниже — точная структура устранивающего разреза и её композиция с AST-1.

5. Теорема не-ретроактивности

Пусть G' — append-only расширение G : все старые узлы и рёбра сохранены побайтно, новые узлы имеют большие birth_rank, а каждое новое ребро направлено от меньшего ранга к большему. Тогда для любого старого узла $v \in V(G)$:

$$Anc_{G'}^*(v) = Anc_G^*(v),$$

и потому

$$OProv_{G'}(v) = OProv_G(v).$$

Доказательство

Новый узел имеет ранг больше ранга каждого старого узла, существовавшего на границе расширения. Путь от нового узла к старому v потребовал бы хотя бы одного ребра от большего ранга к меньшему, что запрещено. Старые пути сохранены, новых путей к старому v нет. Следовательно, его ancestry и, по IADC, provenance неизменны. $\square$

Поздняя запись может создать нового потомка старого узла и унаследовать его provenance. Она не может стать первым источником атома в provenance уже существовавшего корня. Это структурная не-ретроактивность записи, не утверждение о физическом прошлом.

6. Теорема типизированного vertex cut

6.1. Модель блокировки

Фиксируем корень r , сорт σ и атом a . Множество узлов-источников этого атома в ancestry корня:

$$S_{\sigma,a}(r) = \{u \in \text{Anc}_G^*(r) : a \in \text{Intro}_\sigma(u)\}.$$

Разрешённый блок $B \subseteq V \dot{\hookrightarrow} r$ удаляет в аналитической копии графа узлы B вместе со всеми инцидентными рёбрами. Корень защищён и не удаляется. Блокировка не создаёт новых рёбер, не переносит introductions между узлами и не меняет атомы у оставшихся узлов.

Это контрфактуальная операция над моделью маршрутизации provenance. Она не переписывает immutable опубликованный граф. Если конкретная система допускает отключение каналов до вычисления, та же формула описывает такое отключение; без этого дополнительного операционного условия результат остаётся точным утверждением о записанном графе и его аналитической копии.

После блокировки атом считается присутствующим у корня тогда и только тогда, когда в $G - B$ остаётся хотя бы один путь

$$u \rightsquigarrow r \text{ для некоторого } u \in S_{\sigma,a}(r) \dot{\hookrightarrow}.$$

6.2. Cut-теорема

Для любого допустимого $B \subseteq V \dot{\hookrightarrow} r$ эквивалентны:

23. типизированное вхождение (σ, a) отсутствует в provenance корня после блокировки B ;

24. B пересекает каждый путь от каждого intro-source $u \in S_{\sigma,a}(r)$ к r .

Формально:

$$a \notin \text{Prov}_{G-B,\sigma}(r) \Leftrightarrow \forall u \in S_{\sigma,a}(r) \forall P: u \rightsquigarrow r, B \cap V(P) \neq \emptyset.$$

Доказательство

Если B не пересекает некоторый путь $P: u \rightsquigarrow r$ от intro-source, весь путь остаётся в $G - B$. Введение a в u достигает r , поэтому атом присутствует.

Обратно, если B пересекает каждый source-to-root путь, ни одно введение a не достигает r в $G - B$. По IADC других источников появления атома у корня нет. Следовательно, атом отсутствует.

□

Теорема не говорит, что удаление строк из payload устраняет зависимость. Payload не является ребром и не является защищённым introduction. Устраняются именно записанные каналы происхождения в объявленной модели.

6.3. Минимальные разрезы как сертификаты

Определим ранг разреза:

$$\lambda_{G,r}(\sigma, a) = \min \lambda$$

Если допустимых B нет, полагаем $\lambda_{G,r}(\sigma, a) = \infty$.

Любое минимальное B , достигающее λ , является конечным проверяемым сертификатом: достаточно перечислить все source-to-root пути и проверить, что каждый пересечён, а затем убедиться, что разреза меньшей мощности нет.

Граничные случаи фиксируются явно:

- если $S_{\sigma,a}(r) = \emptyset$, атом уже отсутствует и пустой разрез имеет размер 0;
- если сам защищённый корень вводит a , допустимого разреза нет и $\lambda = \infty$: путь длины 0 состоит из r , а $r \notin B$;
- один общий dominator всех путей даёт разрез размера 1;
- две непересекающиеся до корня ветви от разных источников требуют по меньшей мере двух блокируемых узлов.

Размер минимального разреза измеряет только структурную избыточность записанных путей происхождения. Он не измеряет истинность, надёжность, физическую силу или значимость атома.

7. Положительные и отрицательные контроли

Аппарат обязан принимать не только случаи отсутствия. Иначе отрицательный результат можно было бы получить запретом на всё.

Контроль	Что должно произойти в записи	Чего он не удостоверяет
BINDING введён защищённым BINDING-узлом	origin-tagged binding входит в provenance потомка	что запись действительно связывает символ с названным
EVENT_RECORD введён EVENT-узлом	event-record входит в provenance потомка	что событие произошло

Контроль	Что должно произойти в записи	Чего он не удостоверяет
PURPOSE введён GOAL-узлом	goal-record входит в provenance потомка	что цель существует вне записи, оправдана, достигнута или имеет смысл
более поздний VERDICT наследует parents	он получает их origin-tagged provenance	что его payload истинен
два независимых source-to-root пути	минимальный разрез имеет размер 2	физическую независимость двух причин
общий dominator	минимальный разрез имеет размер 1	физическую причинность dominator

Отрицательный spoof-контроль помещает строки binding, event, goal, «ракета полетела» и «purpose» внутрь payload FORMAL-узла. Типизированные множества binding, event, goal при этом остаются пустыми. Имя не внесло названное; строка не создала защищённый kind.

Исполняемый checker проверяет конечную JSON-структуру, точное ancestor-union, origin-теги, prefix commitment и cut-сертификаты. Его valid означает соответствие этому контракту. Оно не означает, что названный объект предъявлен, событие произошло, цель оправдана или какая-либо строка истинна.

8. Композиция с AST-1: различие обязано иметь предка

Пусть K — множество допустимых полных записей с фиксированным идентификатором заключения r . Пусть

$$\rho: K \rightarrow R$$

— base-проекция. В ней зафиксированы все данные, различие которых не должно ошибочно приписываться типизированному каналу: идентификаторы узлов, топология DAG, правила типизации, root id и иные объявленные core-поля.

Определим origin-tagged ancestry-проекцию:

$$\eta_r(k) = \{(\sigma, a, u) : u \in \text{Anc}_{G_k}^*(r), a \in \text{Intro}_{k, \sigma}(u)\}.$$

Пусть записанное заключение вычисляется только из base и этой проекции:

$$C_r(k) = F(\rho(k), \eta_r(k)).$$

И пусть отдельно заданная target-координата $Q: K \rightarrow Y$ действительно воспроизводится заключением:

$$C_r = Q.$$

Тогда по теореме факторизации RBI-F:

$$\ker \rho \cap \ker \eta_r \subseteq \ker Q.$$

Следовательно, для любой base-конфликтующей пары

$$\rho(k_0) = \rho(k_1), Q(k_0) \neq Q(k_1),$$

необходимо

$$\eta_r(k_0) \neq \eta_r(k_1).$$

Разность двух η_r содержит хотя бы один origin-tagged атом (σ, a, u) . Поскольку одинаковая base-проекция уже фиксирует топологию и root, его origin u локализован в ancestry заключения, а

$$b(u) \leq b(r).$$

То есть если заключение различает то, чего не различал base-core, различающий типизированный канал был введён в ancestry заключения. Узел с рангом больше $b(r)$ не может быть первым источником этого различия для уже существующего r .

Cut-теорема делает локализацию операционно точной внутри модели: чтобы устранить из root-provenance конкретный различающий атом, нужно и достаточно перекрыть все пути от всех его intro-sources к корню. Минимальный vertex cut указывает наименьший записанный набор каналов, без которого это различие больше не достигает корня.

Если некоторое различие не локализуется в η_r , но заключение всё же его использует, значит, функция прочитала дополнительный вход, не включённый в объявленную ancestry-проекцию. Назвать этот вход «внутренним смыслом символов» недостаточно: либо его точная координата внесена в readset, либо заявленная факторизация ложна.

9. Жёсткая граница вывода

Из типизированного provenance допустимы только структурные заключения вида:

$$(\sigma, a, u) \in OProv_G(r),$$

«узел u типизированно ввёл запись атома a сорта σ , и в объявленном DAG существует путь $u \rightsquigarrow r$ ».

Запись таких строк в payload не переносит названное в граф; без отдельного канала не следуют:

$$Refers(a, x), Occurs(x), True(a), Causes(x, y), JustifiesPurpose(a).$$

Максимальная уступка: ракета, событие и вопрос «зачем»

1. Уступка принимается целиком

Для этого раздела принимается самый сильный возможный вариант возражения. Допустим одновременно, что:

- ракета существует;
- ракета построена;
- формальная запись была использована при проектировании и управлении;
- объявленный расчёт выполнен без ошибки;
- наблюдаемая траектория совпала с рассчитанной в пределах принятого критерия;
- ракета летит.

Раздел не выводит эти пункты из строк и не предлагает заменить ими названное. Они вводятся как **максимальная уступка** оппоненту. Поэтому спор не может быть остановлен фразой «но ракета действительно полетела»: здесь её полёт уже допущен.

Из этой уступки следует возможность успешного применения по принятому контракту. Из неё не следует, что ракета, полёт, связь расчёта с ракетой или цель полёта находились в более ранних линиях. Позднее наличие не является прежним содержанием; позднее совпадение не является прежней ссылкой; осуществившееся событие не становится предком записи, созданной до него.

Пусть r_0 — неизменённый ранний формальный record, а $r_1 \geq r_0$ дополнительно содержит binding-, application-, event- и telemetry-records. Даже при принятой уступке выполняется тождество раннего носителя:

$$bytes_{r_1}(r_0) = bytes_{r_0}(r_0).$$

Ни полёт, ни наблюдение, ни успех не дописывают себя назад в r_0 . Они могут войти только в новую доступную запись и создать новый этап вывода. Формула могла участвовать в создании ракеты; это не означает, что ракета находилась в формуле.

2. Четыре разных отношения

Ни одно написанное ниже имя не получает названное им одним фактом написания. Символ e остаётся индексом; строка `rocket` остаётся строкой; запись `EVENT` остаётся записью данного типа. Чтобы не переносить свойства между разными областями, фиксируются четыре отдельных предиката:

- $Contains(r, s)$: последовательность s входит в record r
- $Entails_C(\Gamma, \phi)$: по правилам calculus-record C существует принятая derivation-record из Γ к ϕ
- $Occurs_P(e)$: индексу e присвоен occurrence-status по отдельно объявленному protocol P
- $Causes_M(x, e)$: пара (x, e) получает causal-status по отдельно объявленному mechanism/model contract M

Это типизированные отношения с разными условиями входа. В частности, без явного bridge-контракта запрещены переходы

$$\begin{aligned}
 Contains(r, s) &\Rightarrow \not\Rightarrow Entails_C(\Gamma, s), \\
 Contains(r, s) &\Rightarrow \not\Rightarrow Occurs_P(s), \\
 Entails_C(\Gamma, \phi) &\Rightarrow \not\Rightarrow Occurs_P(\phi), \\
 Occurs_P(e) &\Rightarrow \not\Rightarrow Causes_M(r, e), \\
 Causes_M(x, e) &\Rightarrow \not\Rightarrow Contains(x, e).
 \end{aligned}$$

Обратные импликации также не предоставляются автоматически. Реальное либо уступленное осуществление события не делает любую строку с его именем доказательством события. Принятый causal-status не делает причину буквальной подстрокой результата. Формальная выводимость не создаёт occurrence-status. Буквальное вхождение строки не создаёт ни выводимость, ни событие, ни причинность.

Даже запись вида

$$\ulcorner Occurs_P(e) \urcorner$$

есть запись утверждения внутри настоящего текста. Она не является тем, чему присвоен индекс e , и не осуществляет это. Чтобы результат не получил скрытую привилегию, то же ограничение применяется к словам ракета, существует, летит, причина, цель И смысл во всём разделе.

3. Что именно даёт успешный полёт

Максимальная уступка разрешает построить расширенную запись

$$r^+ = (r_0, b, a, o, p, m),$$

где b — binding-record, a — application-record, o — observation/event-record, p — protocol-record, a — mechanism/model-record. На этом расширенном входе могут быть установлены, например:

- совпадение формального output-record с measurement-record по объявленной метрике;
- наличие полного binding между полями модели и полями application-record;
- occurrence-status по P ;

- causal-status по M ;
- достижение заранее записанного performance criterion.

Каждый такой результат условен относительно добавленных записей и контрактов. Он не превращает

$$r_0$$

в

$$(r_0, b, a, o, p, m).$$

Именно поэтому технический успех не опровергает границу. Он показывает, что после внесения binding, механизмов, наблюдений и критериев совместная система способна получить новый статус. Он не показывает, что прежние линии автономно выбрали эти добавления.

4. Первый момент определимости

Пусть K — конечное множество допустимых полных records, $V_t: K \rightarrow X_t$ — запись, доступная на объявленном этапе t , а $A(k) \subseteq D$ — непустое множество допустимых answer-codes для target k . Zero-error selector на этапе t существует тогда и только тогда, когда в каждом волокне доступного view есть общий допустимый ответ:

$$\forall x \in V_t(K): \bigcap_{k: V_t(k)=x} A(k) \neq \emptyset.$$

Если два полных records имеют один view, но требуют непересекающиеся множества ответов, никакой детерминированный zero-error selector этого view не решит оба случая. Рандомизация не устраняет запрет zero-error: её support также должен лежать в указанном пересечении. Поздний канал может разделить прежнее волокно; тогда определимость возникает впервые на новом этапе, а не обнаруживается задним числом в старом.

FDT говорит только об объявленной конечной фильтрации и answer-codes. Она не удостоверяет, что view исчерпывает мир, что код является референтом или что принятый target правилен.

5. Горизонт 100 / 200 / 300 лет

Вопрос «зачем?» фиксируется не риторически, а как конечный тест двух допустимых продолжений. Возьмём $k_A, k_B \in K$ с разными допустимыми goal-codes:

$$A(k_A) = \{g_A\}, A(k_B) = \{g_B\}, g_A \neq g_B.$$

Объявим конечную фильтрацию:

Этап	Доступные каналы
T_0	formal core
T_1	formal core + application binding
T_2	предыдущие записи + event/telemetry record
T_{100}	предыдущие записи + horizon-record 100 years
T_{200}	предыдущие записи + horizon-record 200 years
T_{300}	предыдущие записи + horizon-record 300 years
T_G	предыдущие записи + goal-channel

Пусть views k_A и k_B совпадают на всех этапах от T_0 до T_{300} . Тогда каждое соответствующее волокно содержит оба records, а

$$A(k_A) \cap A(k_B) = \emptyset.$$

Следовательно, до T_G zero-error выбор между g_A и g_B невозможен. Ни успешное применение, ни event-record, ни добавление одинаковых строк 100, 200, 300 не разделяют эту пару. На T_G goal-channel может записать g_A в одном продолжении и g_B в другом; views расходятся, и selector становится возможен.

Этот результат конечен и точен. Он не утверждает, что цель не может быть добавлена после 300 лет, что различие никогда не возникнет, что у кого-либо «нет смысла» или что бесконечное будущее исчерпано. Он устанавливает только следующее: в объявленной фильтрации до goal-channel прежний доступный record не выбирает конфликтующие goal-codes; появившийся goal-record является новым различающим входом.

Запись цели всё ещё не является её оправданием. Факт появления строки g_A устанавливает наличие goal-code в record по контракту; он не заставляет принять g_A , не делает её подходящей и не отвечает, почему она должна быть целью.

6. Четыре target вместо одного слова «цель»

Слово цель часто скрывает четыре независимых вопроса. Они должны проверяться отдельно:

Target	Точная координата	Чего она сама не устанавливает
goal	какой goal-code записан	принят ли он кем-либо

Target	Точная координата	Чего она сама не устанавливает
endorsement	какой endorsement-status записан и кем по контракту	соответствует ли цель ситуации
fit	какой fit-status получен по объявленному criterion-record	почему критерий следует принять
justification	какой justification-status присвоен по отдельному нормативному контракту	наличие события или причинной связи вне этого контракта

Формально это четыре функции, а не одно поле:

$$Q_G: K \rightarrow G, Q_E: K \rightarrow E, Q_F: K \rightarrow F, Q_J: K \rightarrow J.$$

Даже если Q_G уже определима, можно сохранить одинаковыми все доступные записи и построить два допустимых продолжения с разными Q_E . После добавления endorsement-record остаётся возможность различия по Q_F ; после fit-record — по Q_J . Для каждого target условие FDT применяется заново. Нельзя закрыть четыре волокна одним написанным словом.

Поэтому возможна строгая последовательность:

$$\text{goal-record} < \text{endorsement-record} < \text{fit-record} < \text{justification-record},$$

где знак $<$ означает только порядок добавления в конкретном протоколе, а не логическую или моральную иерархию. Ни один шаг не возникает из предыдущего без указанного bridge.

7. Перестановка binding как частный случай

Пусть один и тот же carrier c допускает два binding-record:

$$\beta_A: c \mapsto \text{ref} - A, \beta_B: c \mapsto \text{ref} - B.$$

До чтения binding-channel:

$$V_0(k_A) = V_0(k_B) = c,$$

но допустимые singleton answers различны. По FDT zero-error selector из одного c отсутствует. После чтения binding-channel

$$V_1(k_A) = (c, \beta_A), V_1(k_B) = (c, \beta_B),$$

и волокно разделяется. Это relation-valued форма того же предела, который RBI выражает как невозможность факторизации target через проекцию carrier. Binding не был извлечён из одинаковых линий: он был внесён отдельной различающей записью.

Перестановка особенно важна потому, что carrier можно оставить побайтно неизменным. Замена ref-A на ref-B происходит не внутри него, а в координате, к которой ранний view не имел доступа. Наличие в carrier строк A, B, object, rocket или даже полной сериализации обоих bindings не выбирает один из них: Contains не становится binding-функцией.

8. Transcript- и sensor-controls

Интерактивность не отменяет предел; она делает добавленный канал видимым. Пусть начальный view двух records одинаков:

$$V_0(k_A) = V_0(k_B) = c.$$

После запроса система получает разные transcript-records или sensor-records:

$$V_1(k_A) = (c, t_A), V_1(k_B) = (c, t_B), t_A \neq t_B.$$

Если t_A, t_B разделяют все конфликтующие answer-sets, selector на расширенном view возможен. Это положительный control FDT: новый ответ возникает там, где появился различающий input. Он не доказывает, что прежний carrier уже содержал transcript, показание, событие или референт.

Sensor-record также остаётся record. Его принятие как measurement, его связь с устройством, классификация события и надёжность канала требуют отдельных protocol-, binding- и criterion-records. Если carrier и полный transcript/sensor view одинаковы, а допустимые targets всё ещё различаются, даже интерактивная процедура не имеет zero-error selector на этом этапе. Тогда нужен ещё один различающий канал либо честный статус неопределённости.

9. Что остаётся от вопроса «зачем?»

После максимальной уступки больше нельзя отвечать только перечислением технических успехов. Можно принять, что расчёт был точен, binding полон, механизм сработал, запись события прошла протокол и ракета летит. Всё это отвечает на вопросы вида «что было вычислено?», «что было сопоставлено?», «что получило occurrence-status?» и «что получило causal-status по M ?»

Вопрос why распадается как минимум на causal explanation, design intention, selected goal, endorsement и justification. Causal relation может быть записан без цели; цель — без endorsement; endorsement — без fit; fit — без justification. Математический аппарат способен переносить уже введённые goal-, utility-, criterion- и justification-records по своим правилам. Перенос не является автономным выбором входа и не превращает вход в его оправдание.

Точный итог поэтому сильнее расплывчатого спора о полезности и уже ложного тотального заявления об отсутствии смысла:

Даже если ракета существует, построена, рассчитана, совпала с расчётом и летит, этот успех не помещает ракету, полёт, binding или цель в более ранние линии. До различающего канала конфликтующие targets не выбираются с zero-error; после канала появляется новая запись, а не доказательство того, что названное всегда находилось в прежней.

Это не ожидание будущего объяснения. Для каждого зафиксированного этапа это законченный отрицательный результат о функции данного входа. Если позднее будет добавлен новый канал, возникнет новый вход и новый вопрос. Ранний record от этого не изменится.

04. КРАСНАЯ КОМАНДА: ГДЕ ИМЕННО МОЖНО СЛОМАТЬ PHASE 2

Назначение этого файла

Здесь исследование атакует собственные самые сильные фразы. Цель — не смягчить вывод, а удалить из него всё, что держится на незаявленном чтении, подмене носителя референтом или риторической скорости. После атаки должны остаться только утверждения, для которых указаны входы, область действия и точный контрпример, способный их опровергнуть.

Исходное ограничение не отменяется ни в одном разделе:

Слово не является своим референтом. Совпадение строк не есть совпадение названного; различие строк не есть различие названного. Запись слова «количество», «ракета», «полет», «цель» или «Марс» вносит в документ только различимые знаки. Всё остальное требует отдельно объявленного отношения.

Поэтому ниже не используются рассказы о внешних объектах как доказательства теорем. Внешние работы и предшественники также не являются посылками. Приоритет и новизна имеют статус NOT ASSESSED.

1. Четыре отношения, которые нельзя склеивать

Для одного и того же носителя x необходимо различать как минимум четыре предиката:

- $\text{Contains}(x, s)$: в носителе имеется строка или код s ;
- $\text{Entails_C}(x, p)$: из x следует код p относительно явно заданного исчисления c ;
- $\text{Occurs_P}(e)$: индексу e присвоен occurrence-status по отдельно объявленному протоколу P ;
- $\text{Causes_M}(x, e)$: x является причиной события e относительно причинной модели M .

Для графа происхождения используется ещё один, структурный предикат $\text{InProv_G}(r, a)$: типизированный atom-code a достигает корня r по записанным рёбрам графа G . Он намеренно не называется occurrence.

Ни одна стрелка между этими отношениями не принимается автоматически:

Contains не даёт Entails_C ; Entails_C не даёт Occurs_P ; InProv_G не даёт ни Occurs_P , ни Causes_M ; Occurs_P не даёт Causes_M ; Causes_M не превращает имя события в событие. Даже если все утверждения записаны рядом, это строки до тех пор, пока мосты между ними не объявлены и не проверены в собственной области.

Это сразу блокирует четыре типовые подмены:

25. Строка EVENT_RECORD не доказывает, что событие произошло.
26. Строка GOAL не доказывает существование, принятие, достижение или оправданность цели.
27. Формальный условный вывод не доказывает, что его условия выполнены вне исчисления.
28. Ребро в provenance-графе не доказывает физическую причинность.
-

2. Точная мишень AST-1 и атаки на нее

2.1. Ядро утверждения

Пусть K — объявленное множество записей; $\rho: K \rightarrow R$ — объявленная проекция core; $\eta_v: K \rightarrow E$ — объявленная типизированная, origin-tagged проекция рефлексивной ancestry узла заключения v ; $Q: K \rightarrow A$ — фиксированная целевая классификация. Если существует функция F такая, что

$Q(k) = F(\rho(k), \eta_v(k))$ для каждого k из K ,

то

$\ker(\rho) \cap \ker(\eta_v) \subseteq \ker(Q)$.

Эквивалентная контрапозиция: если $\rho(k_0) = \rho(k_1)$, но $Q(k_0) \neq Q(k_1)$, то $\eta_v(k_0) \neq \eta_v(k_1)$. При определении η_v как объединения origin-tagged введений по ancestry это означает: различающий типизированный атом был введен некоторым предком v . Более поздний узел не может быть его первым источником в уже зафиксированной ancestry.

Теорема говорит о функциях и объявленных кодах. Она не говорит, что код $Q(k)$ является референтом, что ancestry является физической историей или что ρ исчерпывает всё доступное наблюдение.

2.2. Атака: одинаковые символы, разные референты

Возражение. Две одинаковые строки могут употребляться о разном. Значит, равенство $\rho(k_0) = \rho(k_1)$ ничего не говорит о референтах.

Вердикт. Возражение верно и не ломает AST-1. Оно ломает расширенную фразу «одинаковая запись означает одинаковый предмет». Такой фразы здесь нет. AST-1 устанавливает лишь невозможность вычислить различающий код Q через одинаковые значения обеих объявленных проекций.

Repair. Везде писать «одинаковые значения объявленной проекции», а не «одинаковые миры» и не «тот же предмет». Если Q задуман как семантический статус, мост от кода к статусу остается отдельной, не проверенной машиной посылкой.

2.3. Атака: разные символы, тот же референт

Возражение. Различие $\text{eta_v}(k_0) \neq \text{eta_v}(k_1)$ может быть орфографическим или кодировочным; оно не обязано различать названное.

Вердикт. Верно. Различающий atom-code необходим для объявленной кодовой классификации, но сам по себе не достаточен для различия референтов.

Repair. Вывод AST-1 заканчивается на локализации различающего origin-tagged кода. Любая фраза «следовательно, различаются объекты» требует отдельного контракта эквивалентности и отдельной проверки.

2.4. Атака: скрытый readset, oracle или внешняя память

Возражение. Реальный вычислитель может читать часы, сеть, датчик, память оператора, системный prompt или oracle, хотя эти каналы отсутствуют в ρ и eta_v . Тогда он различит записи с одинаковыми объявленными входами.

Вердикт. Это настоящий контрпример к широкой фразе «никакой процесс не сможет различить». Это не контрпример к AST-1, потому что такой процесс не является функцией только от $(\rho, \text{eta_v})$.

Repair. Либо включить каждый читаемый канал в проекцию, либо поставить статус DECLARED-VIEW-ONLY-NOT-EXHAUSTIVENESS-ATTESTED. Нельзя одновременно пользоваться скрытым oracle и утверждать, что различие возникло из core.

2.5. Атака: топология и правила меняются между записями

Возражение. Один и тот же набор typed atoms может обрабатываться разными правилами или в разных DAG.

Вердикт. Если топология, правила или идентификатор заключения влияют на Q , но не входят в ρ , посылка факторизации не установлена.

Repair. Зафиксировать их глобально либо включить в ρ . AST-1 нельзя применять к невидимо меняющемуся исчислению.

2.6. Строгий falsifier AST-1

AST-1 опровергнут внутри своей заявленной модели, если предъявлены $K, \rho, \text{eta_v}, Q, F, k_0, k_1$, для которых одновременно истинны:

29. $Q(k) = F(\rho(k), \text{eta_v}(k))$ для всех k ;

30. $\rho(k_0) = \rho(k_1)$;

31. $\text{eta_v}(k_0) = \text{eta_v}(k_1)$;

32. $Q(k_0) \neq Q(k_1)$.

Рассказ о скрытом канале, иной кодировке или внешнем референте не удовлетворяет этому falsifier: он меняет модель или одну из посылок.

3. Точная мишень FDT и атаки на нее

3.1. Критерий первого момента определимости

На стадии t каждой записи k сопоставлен объявленный view $v_t(k)$ и непустое множество допустимых answer-codes $A(k)$. Детерминированный zero-error селектор s_t существует тогда и только тогда, когда для каждого fiber $B = v_t^{-1}(v)$ выполнено

$\text{intersection } \{ A(k) : k \in B \} \neq \text{empty}.$

Если последующие views только уточняют предыдущие, существование селектора монотонно: после первой selectable-стадии оно не исчезает. В конечной объявленной фильтрации можно указать первый такой индекс. Это FDT. Это критерий для answer-codes в объявленном конечном record-set, не доказательство истинности ответов о мире.

3.2. Атака: произвольный константный селектор

Возражение. Можно всегда напечатать один и тот же ответ, не различая записи.

Вердикт. Да — если этот answer-code входит в $A(k)$ для каждой записи соответствующего fiber. Поэтому тезис «одинаковый view всегда делает ответ невозможным» ложен.

Repair. Блокирует не само равенство views, а пустое общее пересечение допустимых множеств. Константный селектор — обязательный отрицательный control: когда общее пересечение непусто, система должна признать selectability уже на этой стадии.

3.3. Атака: попарных пересечений недостаточно

Возражение. Если каждая пара записей имеет общий допустимый ответ, значит общий ответ существует для всего fiber.

Вердикт. Ложно. Для

- $A_0 = \{a, b\},$
- $A_1 = \{b, c\},$
- $A_2 = \{a, c\}$

все попарные пересечения непусты, а $A_0 \cap A_1 \cap A_2$ пусто.

Repair. Проверять полное пересечение всего fiber. Pairwise-only алгоритм является falsified implementation даже тогда, когда он проходит все двухэлементные примеры.

3.4. Атака: randomized prior и средняя успешность

Возражение. Рандомизация или prior позволяют угадывать лучше.

Вердикт. Они могут увеличить ожидаемую точность, если разрешена ошибка и задан prior. Они не создают zero-error решение при пустом пересечении: поддержка распределения ответов должна лежать одновременно во всех $A(k)$ данного fiber. При пустом пересечении такой поддержки нет.

Для m взаимно несовместимых singleton-ответов в одном fiber никакая одна и та же randomized policy не дает worst-case success выше $1/m$; prior меняет среднее, а не устраняет проигрывающую ветвь.

Repair. Не смешивать три задачи: deterministic zero-error, randomized zero-error и Bayesian expected accuracy. FDT в текущем контракте решает первые две на уровне поддержки; численная средняя точность — другой контракт.

3.5. Атака: интерактивный transcript

Возражение. Агент может задавать вопросы и потому не ограничен исходным carrier.

Вердикт. Интеракция добавляет read channel. Политика после transcript является функцией carrier, запросов и полученных ответов. Если transcripts различаются, они могут разделить fiber; если полные transcripts совпадают, интерактивность сама по себе различия не создала.

Repair. На каждой стадии включать в v_t полный доступный transcript, состояние policy и все ответы oracle, которые реально читаются. Фраза «невозможно даже интерактивно» допустима только для двух записей, неразличимых по полному объявленному transcript/readset.

3.6. Атака: constitutive convention

Возражение. Иногда правило не описывает внешний предмет, а учреждает формальный статус: запись считается числом, ходом или победой именно по конвенции. Тогда символ вместе с правилом действительно достаточен.

Вердикт. Это ломает безусловную фразу «знаки ничего не устанавливают вообще». Внутри учрежденной практики carrier и конвенция могут детерминировать формальный статус. Но это не экспорт к независимому референту, событию или цели: конвенция сама является дополнительным контрактом и read channel.

Repair. Жесткий точный тезис таков: carrier сам по себе не экспортирует референт, occurrence, cause или justification. Синтаксическая детерминация относительно явно принятой конвенции допускается и тем самым показывает, где именно находится добавленная посылка.

3.7. Атака: бесконечное продолжение

Возражение. Если сегодня, через 100, 200 или 300 записанных лет view недостаточен, из этого не следует, что позднее не появится разделяющий канал.

Вердикт. Верно. Конечная неразрешимость не равна вечной неразрешимости. Последующая binding-, event-, transcript- или goal-запись может разделить прежний fiber. Для бесконечной последовательности нельзя переносить вывод на limit-union без отдельного условия, что один и тот же конфликт сохраняется, либо без подходящего условия компактности. Особенно опасна подмена: «на каждой конечной стадии есть какой-то конфликт» не означает «существует одна пара, конфликтующая на всех стадиях».

Repair. Говорить: «до первой объявленной separating-стадии zero-error селектор отсутствует». Для статуса never добавлять: NEVER-SELECTABLE-ON-DECLARED-FILTRATION, а не «никогда ни при каком будущем дополнении».

3.8. Строгий falsifier FDT

FDT опровергнут в объявленной конечной модели, если существует хотя бы один fiber с пустым полным пересечением $\Delta(k)$, но zero-error селектор на нем существует; либо если все fiber-intersections непусты, но ни один селектор выбрать нельзя. Проверка только попарных пересечений не считается проверкой FDT.

4. Provenance и cut: что именно они доказывают

4.1. Provenance — не причинность и не семантика

В typed-provenance DAG каждый узел имеет строгий birth_rank, родителей только меньшего ранга, допустимый по kind набор локальных введений и declared provenance, равную наименьшему объединению локальных введений по рефлексивной ancestry. Origin-tag сохраняет узел первого введения атома.

Из этого следует record-факт: если typed atom присутствует в provenance корня, его ввел корень или один из его предков. Не следует:

- что событие с таким именем произошло;
- что binding соединил код с внешним предметом;
- что goal был чьим-либо намерением;
- что ребро было причиной физического результата;
- что payload-строка с тем же словом является typed introduction.

Payload намеренно инертен. Запись "goal": "Mars" внутри payload не вводит typed goal, если защищенное поле introduces.goal пусто. И наоборот, typed goal остается кодом, а не самой целью.

4.2. Mutable history, backdating и циклы

Возражение. Можно переписать старого родителя, присвоить новому узлу ранний номер, добавить обратное ребро или образовать цикл.

Вердикт. Это не опровержение теоремы для immutable DAG; это выход из ее области. Но это опровергает риторическую фразу «позднее никогда нельзя изменить прошлую provenance» без зафиксированного контракта истории.

Repair. Строго разделять:

- declared birth_rank и засвидетельствованное физическое время;
- append-only запись и редактируемый файл;
- DAG и циклическую систему;
- сохранение exact prefix и доверие к одной строке freeze_status.

Машина проверяет порядок полей и хэш объявленного prefix. Она не проверяет, что автор не создал весь файл задним числом. Backdated rank остается backdated строкой, если нет отдельного внешнего свидетельства времени.

Для циклов требуется иная семантика, например отдельно заданная least-fixed-point конструкция. Текущая ancestor-union теорема на циклы не переносится молча.

4.3. Hash collision и статус commitment

Равенство SHA-256 digests не является логическим равенством произвольных байтов: возможность коллизии не исключена теоремой. Поэтому commitment имеет два уровня:

33. внутри исполнения checker пересчитывает digest канонического prefix и сравнивает строки;
34. утверждение, что этот digest однозначно фиксирует внешний исторический объект, зависит от криптографического допущения и канала публикации.

Repair. Для чисто логического утверждения хранить и сравнивать exact canonical bytes. Для практического commitment явно назвать алгоритм и допущение collision resistance. Не превращать sha256:... в машину времени.

4.4. Cut-теорема

Для фиксированного typed atom a у корня r и политики удаления узлов без rewiring множество удаляемых вершин D исключает r . После пересчета least provenance атом a исчезает у r тогда и только тогда, когда D пересекает каждый путь от любого introducer a к r .

Минимальный cut — минимальная по мощности такая hitting set. Его мощность показывает структурную избыточность путей записи, не важность, истинность или причинную силу атома.

Строгий falsifier. При сохранении всех посылок предъявить одно из двух:

- α пересекает каждый introducer-to-root path, но α остается после пересчета;
- существует surviving introducer-to-root path, но α исчезает.

Удаление payload-текста, изменение правил, rewiring родителей, удаление самого защищенного корня или создание нового introducer после удаления — другие операции и не являются falsifier текущей cut-теоремы.

5. Самые опасные семантические расширения

5.1. Условное следование против occurrence

Формула может корректно давать y при условии x . Наличие доказательства $x \rightarrow y$ не вносит occurrence x , occurrence y или случай мира, к которому они якобы относятся. Даже машинно проверенный derivation подтверждает только Entails_C при зафиксированных правилах C .

Запрещенная фраза: «расчет доказал, что ракета полетит».

Точный repair: «в исчислении C , при assignment v и предпосылках n , получен output-code o ». Отдельно нужны binding, условия применения, event-channel и критерий соответствия. Слова rocket, flight и Mars не совершают ни полета, ни привязки.

5.2. Goal против justification

Даже корректно добавленный goal-code решает только задачу выбора среди goal-codes в объявленной relation. Он не доказывает:

- что это действительное намерение автора или группы;
- что действие достигает названного;
- что цель оправданна;
- что результат имеет смысл через 100/200/300 лет;
- что сам вопрос «для чего?» имеет единственный допустимый ответ.

Goal может сделать FDT selectable потому, что target relation определена на goal-codes. Justification требует другой relation, других acceptable sets и других каналов. Нельзя переименовать goal-A в «смысл» и считать переход выполненным.

5.3. Binding против референта

Binding-record может различить два record-cases и тем самым позволить selector. Он всё еще является записью. Чтобы утверждать, что binding достигает внешнего объекта, нужен operational bridge: процедура доступа, критерии сохранения идентичности и условия ошибки. AST-1 локализует, где в record ancestry впервые появился различающий binding-code; она не превращает его в названное.

5.4. Causation против provenance

Все пути в provenance могут проходить через один узел, и его удаление может убрать atom у корня. Из этого не следует, что тот же узел — физическая причина внешнего события. Cut сертификат относится к переносу типизированной записи по declared graph. Causal claim требует м, интервенционную операцию и связь м с наблюдением; их имена не являются их реализацией.

6. Контроли, без которых результат недостоверен

Control	Что обязана показать система	Какой широкий тезис он ломает
Одинаковый carrier, разные singleton-ответы	До separating channel selector отсутствует	«Любая строка уже несет единственный ответ»
Разные carriers, один общий acceptable answer	Различение не требуется	«Любое различие знаков обязательно значимо»
Непустое общее пересечение fiber	Константный selector принимается	«Одинаковый view всегда блокирует ответ»
Тройка {a,b}, {b,c}, {a,c}	Pairwise overlap не проходит как full overlap	«Достаточно проверить пары»
Полный различающий transcript	Первая selectability появляется на transcript-стадии	«Интеракция ничего не добавляет»
Один и тот же transcript при разных ответах	Интерактивный selector остается невозможным	«Сам факт диалога гарантирует знание»
Formal → application → event → goal	Goal-target становится selectable только на goal-channel	«Расчет или полет уже отвечает, для чего»
Payload-spoof	Слова в payload не вводят typed atom	«Имя создало названное»
Поздний verdict	Не входит в ancestry защищенного старого узла	«Поздняя подпись была посылкой раннего вывода»
Один путь	Минимальный cut имеет мощность 1	Базовая ошибка перечисления путей

Control	Что обязана показать система	Какой широкий тезис он ломает
Два vertex-disjoint пути	Минимальный cut имеет мощность 2	«Достаточно удалить любую одну вершину»
Общий dominator двух путей	Минимальный cut снова имеет мощность 1	«Число путей равно размеру cut»
Неполный cut	Atom сохраняется по surviving path	«Частичное удаление уничтожает provenance»
Удаление только payload-слова	Typed provenance не меняется	«Текст payload был зависимостью»

Control, который система не способна принять, важнее удобного положительного примера: он показывает, не защит ли желаемый вывод в название поля.

7. Самоприменение: этот документ не исключение

Этот файл — carrier. Его буквы не являются AST-1, FDT, референтом, событием, целью, смыслом или опровержением. Они становятся формулами только относительно читателя, синтаксиса, кодировки и объявленной метатеории. Если читатель не принимает эти правила, наличие строк не заставляет его принять названное.

Использование множеств, функций, равенства, логики и языка не является круговым доказательством «математика истинна о мире». Исследование делает более узкую операцию: внутри явно названной метатеории выводит границу экспорта из carrier. Оно может проверить собственную условную форму, но не может из собственных символов произвести внешний референт. Поэтому самоприменение не освобождает текст от его же ограничения — оно подчиняет ему текст.

Возможное обвинение в performative contradiction было бы успешным только если документ утверждал бы одновременно:

- 35. никакие знаки не допускают никаких синтаксических следствий ни при каких правилах;
- 36. следующие знаки синтаксически доказывают это утверждение.

Такого тезиса здесь нет. Допускается формальное следование внутри объявленных правил; отрицается автоматический экспорт от него к референту, occurrence, cause, goal и justification.

Предел самопроверки фиксируется явно:

- PUBLIC-TITLE: UNBOUNDED-STANDALONE — заголовок не является машинно доказанной теоремой;
- AST-1 и FDT: TEXTUAL-THEOREM-IN-DECLARED-METATHEORY;

- выводы checker: факты о предоставленных байтах и declared contracts;
- semantic_status: NOT-MACHINE-VERIFIED;
- полнота readset: DECLARED-VIEW-ONLY-NOT-EXHAUSTIVENESS-ATTESTED;
- соответствие answer-codes референтам: ANSWER-CLASS-CODES-ONLY-NOT-REFERENT-VERIFIED;
- freeze до исполнения: FREEZE-LABEL-ONLY-NOT-MACHINE-VERIFIED;
- исторический приоритет и новизна: NOT ASSESSED.

Успешный checker-output не доказывает мир. Он доказывает, что байты прошли алгоритм с указанным контрактом. Имя VALID-TYPED-PROVENANCE-DAG не является действительной историей; имя VERIFIED-FIRST-DETERMINACY-STAGE не является истинным ответом; строка NOT-MACHINE-VERIFIED не является гарантией честности автора. Каждый статус остается записью, а его применение ограничено объявленным отношением.

8. Финальный реестр: claim / status / falsifier

Claim	Status	Точный falsifier или условие снятия
Если $Q = F(\rho, \eta_v)$, то $\ker(\rho) \cap \ker(\eta_v) \subseteq \ker(Q)$	AST-1 · TEXTUAL-THEOREM-IN-DECLARED-METATHEORY	Два k с одинаковыми ρ и η_v , разными Q , при тотальной проверенной факторизации $Q = F(\rho, \eta_v)$
Core-conflict требует различия typed ancestry, если заключение вычисляется только из core и этой ancestry	AST-1-COROLLARY · CONDITIONAL	Сохранить все ссылки и предъявить core-conflict при равной η_v ; скрытый oracle снимает ссылку, но не опровергает теорему
Различающий origin-tagged atom имеет introducer в рефлексивной ancestry заключения	ANCESTRAL-LOCALIZATION · DECLARED-DAG-ONLY	Валидный по контракту DAG, где atom входит в ancestor-union без introducer среди предков или корня
Zero-error selector на стадии существует iff каждое полное fiber-intersection acceptable sets непусто	FDT · EXACT-FINITE-RELATION-THEOREM	Fiber с пустым полным пересечением и zero-error selector либо все пересечения непусты, но selector невозможен
При refinement selectability монотонна и имеет первый индекс в конечном списке стадий	FDT-FIRST-STAGE · DECLARED-FINITE-FILTRATION	Поздний view является refinement, однако ранее существовавший selector перестает быть корректным при неизменных $A(k)$; либо конечное непустое множество selectable indices не имеет минимума

Claim	Status	Точный falsifier или условие снятия
Рандомизация не дает zero-error при пустом fiber-intersection	FDT-RANDOMIZED · SUPPORT-LEVEL	Распределение с непустой поддержкой, целиком лежащей во всех $A(k)$ пусто-пересекающегося fiber
Отсутствие selector до стадии t означает отсутствие навсегда	REJECTED-OVERREACH	Любая более поздняя separating-стадия; статус допустим только как NEVER-SELECTABLE-ON-DECLARED-FILTRATION
Declared provenance равна least union типизированных введений по reflexive ancestry	PROVENANCE · EXACT-CONTRACT-INVARIANT	Запись, принятая как valid по контракту, но с declared provenance, отличной от вычисленного ancestor-union
Поздний узел не входит в зафиксированную ancestry защищенного раннего узла	NONRETROACTIVITY · IMMUTABLE-DAG/PREFIX-CONDITIONAL	При неизменных protected nodes, strict ranks, parent edges и exact prefix поздний узел становится их предком; rewrite/backdating снимает посылки
Cut удаляет atom у root iff пересекает все introducer-to-root paths	CUT · EXACT-DELETION-CONTRACT	Все пути пересечены, atom остается; либо surviving path существует, atom исчезает — при удалении без rewiring и пересчете least closure
Minimum cut измеряет истинность или физическую причинность	REJECTED-CATEGORY-ERROR	Не требуется: claim не следует из record-теоремы и не принимается
Наличие строки EVENT_RECORD доказывает событие	REJECTED-WORD/REFERENT-SUBSTITUTION	Не требуется: строка устанавливает только Contains; occurrence требует отдельного отношения
Goal-code является целью, смыслом или оправданием	REJECTED-WORD/REFERENT-SUBSTITUTION	Не требуется: goal-selection и justification — разные relations
Binding-code сам достигает внешнего референта	NOT-ESTABLISHED	Для повышения статуса нужен отдельно проверенный operational bridge; его название не считается bridge
Checker-output доказывает внешний мир	REJECTED-SCOPE-EXPORT	Checker подтверждает только обработку предоставленной записи по declared contract
SHA-256 digest логически тождествен exact bytes и доказывает дату создания	REJECTED-OVERREACH	Коллизия ломает первое; создание всего record задним числом ломает второе

Claim	Status	Точный falsifier или условие снятия
Заголовок документа является доказанной теоремой	PUBLIC-TITLE • UNBOUNDED-STANDALONE	Для theorem-status заголовок должен быть заменен точной формулой с областью и доказательством
Семантические claims этого файла проверены checker-ом	NOT-MACHINE-VERIFIED	Статус может быть повышен только отдельным договором наблюдения; еще одна строка статуса недостаточна
Исследование установило историческую новизну или приоритет	NOT ASSESSED	Требуется отдельное сравнение; внешние работы не становятся посылками AST-1/FDT ни при каком результате сравнения

Итог красной команды: самый широкий лозунг не получает theorem-status. Теоремы получают более жесткое преимущество — точную точку поражения. Чтобы опровергнуть AST-1, FDT, provenance или cut, недостаточно назвать скрытый мир, заменить код референтом или добавить поздний канал. Нужно сохранить объявленные посылки и предъявить указанный falsifier. Пока этого нет, carrier остается carrier, а любое различие, которым пользуется вывод, должно быть внесено в его объявленный readset и — для typed DAG — иметь предка.