Band IX

Tractatus Principio

—

Über den Unterschied

Band IX

Buch der Lesung

von

C.H.F. Warnke

Inhaltsverzeichnis

Das Zahlenfraktal von 0 bis 9

1. Textbefund

2. Notationstabelle

3. Grundthese

4. Schichtentrennung

5. Markenraum von 0 bis 9

6. Inschrift als Pfad

7. Spur als integrierte Differenz

8. Operatoren als Kantenrollen

9. Rotation und imaginäre Adressierung

10. Projektionen als Lesekanäle

11. Bindungsgatter und Commit

12. Das ¾Eck als endliche Lock-in-Geometrie

13. Gleichheitszeichen als ZTFΔ-Messstelle

14. Normalform als Eigenresonanz

15. Verkürzung und Readressierung

16. Beispiel: 125+38=163

17. Keine AST-Ontologie

18. Minimaler Implementationsvertrag

19. Tests

20. Endformel

21. Spezifikationssatz

Das Zahlenfraktal von 0 bis 9

Eine ZTFΔ-Spezialisierung sichtbarer Zahlenmarken, Operator-Kanten und des ¾Eck-Snaps

Abstract

Das Zahlenfraktal ist keine neue Rechenmechanik neben ZTFΔ. Es ist die Spezialisierung der ZTFΔ-Engine auf sichtbare Zahlenmarken, Operator-Kanten, Rotationsadressierung und das ¾Eck als Gleichheits-Snap.

Der Kernel rechnet nicht mit Zahlenobjekten. Er integriert Unterschied, liest Spur, bindet Projektionen und liest die eingerastete Eigenresonanz.

Zahlen werden darin nicht als Entitäten behandelt. Eine sichtbare Form wie

125

ist kein Zahlenobjekt, sondern der öffentliche Readout einer getakteten Inschrift. Der Träger besteht aus den sichtbaren Marken:

```
M = {0,1,2,3,4,5,6,7,8,9}
```

Die Marken 0 bis 8 sind Schalenmarken. Die Marke 9 besitzt eine Sonderrolle:

$$9 := \kappa$$

Sie ist nicht bloß eine weitere Marke derselben Ebene, sondern die sichtbare Bruchkante und der Zyklusschluss. Operatoren wie

$$+,\ -,\ \times,\ \div,\ i,\ -i,\ =$$

sind keine externen Rechenobjekte, sondern Kantenrollen innerhalb desselben Vollzugs.

Der Grundablauf lautet daher nicht:

Zahl -> Rechnung -> Ergebnis

sondern:

Inschrift

-> Pfad

-> Spur

-> Projektion

-> Bindung

-> ¾Eck-Lock

-> Normalform

-> Readout

Formal:

I

$$-> \gamma_I -> \Sigma(\gamma_I) -> \Pi_i(\Sigma)$$

-> ⋈

-> Lock_¾

-> NF

-> read

Das sichtbare Ergebnis ist nicht der ontologische Kern. Es ist die öffentliche Projektion einer eingerasteten Eigenresonanz.

---

1. Textbefund

Die zugrunde liegenden Texte setzen vier feste Punkte.

Erstens arbeitet ZTFΔ mit Gitter, Kanten, Gate, Rotation, Holonomie und sichtbarer Spur. Kreis, Quadrat und Delta sind dort keine Dinge, sondern Lesungsformen: Schließung, Orientierung und Schnitt.

Zweitens beschreibt Delta-Zero den Takt als elementaren Übergang. Eine Spur entsteht durch integrierte Differenz über einen Pfad. Entscheidend ist die Nicht-Kommutativität von Schritt und Lesungsoperation:

Step ∘ O ≠ O ∘ Step

Die Reihenfolge des Vollzugs ist bedeutungstragend.

Drittens bestimmt das ¾Eck ein Ergebnis nicht als gesuchten Zahlenwert, sondern als Lock-in-Lage mit Drift-Signatur. Das Verfahren arbeitet über einen endlichen Rüttelraum und minimalen Drift, nicht über kontinuierliche Optimierung und nicht über ε-Toleranz.

Viertens kritisiert Tractatus Principio die Verdinglichung von Zahl. Zahl erscheint nicht als ursprüngliche Entität, sondern als Resultat von Unterschied, Relation und Lesung.

Daraus folgt:

Zahl := readΣ

nicht:

Zahl := Ding

---

2. Notationstabelle

Zeichen / Form	Kernel-Lesung	Interface-Lesung
|	primitive Unterscheidung / Bindung	Strich / Trennung
0,...,8	Schalenmarken	sichtbare Ziffernmarken
9	κ, Bruchkante, Zyklusschluss	sichtbare Marke 9
10	1|0, neue Leseschale	sichtbare Form "10"
+	|_+, Operator-Kante	Pluszeichen
-	|_-, Operator-Kante	Minuszeichen
×	|_×, Operator-Kante	Multiplikation
÷	|_÷, Operator-Kante	Division
i	R_90°, Rotationsadresse	imaginäre Markierung
-i	R_-90°, inverse Rotationsadresse	negative imaginäre Markierung
=	Π + ⋈ + Lock_¾ + NF	Gleichheitszeichen
Σ	integrierte Spur	nicht direkt sichtbar
Π_i	Projektion / Lesekanal	abgeleiteter Readout
⋈	Bindungsgatter / Kompatibilität	Passung
Lock_¾	endlicher ¾Eck-Snap	Gleichheitsvollzug
NF	Eigenresonanzlesung	Normalform
read	öffentliche Projektion	sichtbares Ergebnis
≡_Σ	gleiche Spurklasse	rückadressierbare Gleichheit
~	Resonanzvergleich	keine klassische Gleichheit
⇝	fällt unter Lesung auf	Zwischen-Readout

Hinweis zur ersten Tabellenzeile: Das Zeichen "|" ist die primitive Unterscheidung beziehungsweise Bindung.

---

3. Grundthese

Das Zahlenfraktal ist ein getakteter Inschriftsträger.

Eine sichtbare Zahl wie:

125

ist keine Zahl-Entität und kein Knoten in einem Objektbaum. Sie ist der sichtbare Readout eines Vollzugs:

```
1 --T--> 2 --T--> 5
```

Der Pfeil ist keine Besitzrelation und keine Parent-Child-Kante. Er bezeichnet einen Takt:

--T--> := operativer Leseschritt

Damit gilt:

125 := read(Fig(1,2,5))

aber:

125 ≠ Fig(1,2,5)

Die Figur ist der Vollzug. Die sichtbare Ziffernfolge ist dessen Interface-Readout.

Deutsch gesagt: Die Zahl steht nicht im Kern. Sie erscheint an der Oberfläche.

---

4. Schichtentrennung

Das Modell trennt drei Ebenen.

4.1 Carrier layer

Der Carrier ist der getaktete Vollzug von Unterscheidungen:

$$T : S_n -> S_(n+1)$$

Er enthält keine Zahlenobjekte.

4.2 Inscription layer

Die Inschrift ist die integrierte Spur:

$$\Sigma_(n+1) = \Sigma_n + H_n$$

mit:

H_n := χ_vise_n · χ_gatee_n · we_n

und:

χ_vis, χ_gate ∈ {0,1}

Dabei ist we_n kein Zahlenwert im klassischen Sinn. Es ist die Differenzlesung der aktuellen Kante unter aktueller Leselage.

Die binären Faktoren sind keine frei erfundenen Gewichtungen. Sie sind Rejektoren:

$$\chi_vis = 0 => H_n = 0$$

$$\chi_gate = 0 => H_n = 0$$

χ_vis = χ_gate = 1 => H_n = we_n

Das Gate ist damit AND-only. Es lässt zu oder verwirft. Es optimiert nicht.

4.3 Interpretation layer

Die sichtbaren Zeichen:

`0,1,2,3,4,5,6,7,8,9,+,-,×,÷,i,-i,=`

sind Interface-Marken.

Sie dürfen gelesen werden, aber sie dürfen nicht als Kernel-Entitäten behandelt werden.

Formal:

Label := readΣ

Ein Label ist gültig nur dann, wenn es verlustlos auf die zugrunde liegende Spurklasse zurückgeführt werden kann:

valid(Label) iff recover(Label) ≡_Σ Σ

Das bedeutet: Rücklesbarkeit verlangt nicht bitgenaue Identität derselben Spurform, sondern Gleichheit der Spurklasse beziehungsweise Readressierungsklasse.

---

5. Markenraum von 0 bis 9

Der sichtbare Markenraum lautet:

```
M = {0,1,2,3,4,5,6,7,8,9}
```

Die Marken 0 bis 8 sind Schalenmarken:

0,...,8 := Schalenmarken

Die Marke 9 besitzt eine Sonderrolle:

$$9 := \kappa$$

Sie ist Bruchkante, Zyklusschluss und Übergang zur nächsten Leseschale.

Daher gilt nicht:

9 ≡ weitere Einheit derselben Ebene

sondern:

9 ≡ Bruchkante / Kantenquerung / Ebenenabschluss

Eine Dezimalfolge wird nicht zuerst als positionaler Wert im klassischen Sinn gelesen, sondern als getaktete Folge sichtbarer Marken:

125 := (1,2,5)

und:

```
125 = read(1 --T--> 2 --T--> 5)
```

Die öffentliche Dezimalarithmetik kann als stabile Projektion erscheinen. Sie ist aber nicht die ontologische Grundoperation.

---

6. Inschrift als Pfad

Eine Fragestellung ist zunächst eine Inschrift:

```
I = (m_0, m_1, ..., m_(N-1))
```

mit:

$$m_n \in 0,1,2,3,4,5,6,7,8,9,+,-,\times,\div,i,-i,=$$

Diese Marken sind keine Dinge. Sie sind Lesemarken innerhalb eines Vollzugs.

Die Inschrift wird als Pfad gelesen:

I -> γ_I

Der Pfad besteht aus getakteten Übergängen:

$$\gamma_I = (e_0, e_1, \ldots, e_(N-1))$$

Jede Kante ist ein Vollzug:

$$e_n : x_n -> x_(n+1)$$

und jeder Vollzug steht unter einer Lesungsoperation:

T_n := Step ∘ O_n

Dabei ist O_n je nach Marke eine Ziffernlesung, Operator-Kante, Bruchkante, Rotation oder Snap-Aktivierung.

Wichtig:

Step ∘ O ≠ O ∘ Step

Der Vollzug ist also geordnet. Das Zahlenfraktal ist kein Zustandsautomat. Es ist ein Pfad-Integrator.

---

7. Spur als integrierte Differenz

Jeder Schritt trägt eine Schrittladung:

δ_n = we_n

Diese Ladung ist kein Zahlenwert im klassischen Sinn. Sie ist der relationale Effekt der Kante im aktuellen Vollzug.

Die Spur des Pfades lautet:

Σγ_I = Σ_(n<N) H_n

mit:

H_n := χ_vise_n · χ_gatee_n · we_n

Äquivalent, wenn Sichtbarkeit und Gate beide aktiv sind:

χ_vise_n = χ_gatee_n = 1 => H_n = we_n

Die bisherige Schreibweise:

$$\Sigma_(n+1) = \Sigma_n + H_n$$

ist die Zahlenfraktal-Schreibweise der ZTFΔ-Spurakkumulation.

Wenn nichts sichtbar ist, entsteht keine Spur:

$$\chi_vis \equiv 0 => \Sigma = 0$$

Wenn das Gate geschlossen ist, entsteht ebenfalls keine Spur:

$$\chi_gate \equiv 0 => \Sigma = 0$$

Persistenz entsteht nur dort, wo die integrierte Differenz nicht verschwindet:

$$\Sigma(\gamma) \neq 0$$

---

8. Operatoren als Kantenrollen

Operatoren sind keine externen Rechenzeichen. Sie sind Kantenrollen.

$$+ := |_{+}$$

- := |_-

$$\times := |_{\times}$$

$$\div := |_{\div}$$

$$=:= |_{=}$$

Eine Inschrift wie:

$$125 + 38 =$$

ist daher nicht primär als abstrakter Syntaxbaum zu lesen:

Add(125,38)

sondern als getaktete Inschrift:

```
1 --T--> 2 --T--> 5 --|_+/T--> 3 --T--> 8 --|_=/T--> Snap
```

Der Operator erzeugt keine Wertaddition im Kernel. Er markiert eine Expansionskante innerhalb der Inschrift.

Klassische Addition kann als stabile Projektion des Ergebnisses erscheinen. Sie ist aber nicht die Grundbewegung.

---

9. Rotation und imaginäre Adressierung

Tiefe wird im Zahlenfraktal nicht als zusätzliches Zustandsfeld gespeichert.

Sie wird über Rotation adressiert:

$$i := R_{90^\circ}$$

$$i^2 := R_1 80^\circ$$

$$i^3 := R_2 70^\circ$$

$$i^4 := id$$

Damit ist i keine imaginäre Entität. i ist eine Rotationsrolle der Lesung.

Eine Form wie:

i125

bedeutet nicht "imaginäre Zahl" im klassischen Sinn, sondern:

i125 := read_R90(Fig(125))

also dieselbe Figur unter gekippter Lesung.

Tiefe ist daher kein Feld im Kernelzustand:

depth notin KernelState

sondern eine rekonstruierbare Rolle der Spur:

depth := read_dΣ

---

10. Projektionen als Lesekanäle

Die integrierte Spur ist noch kein öffentliches Ergebnis. Sie wird durch Projektionen gelesen:

```
a_i = Π_i(Σ)
```

Diese Projektionen sind Lesekanäle.

Im Zahlenfraktal gibt es mindestens folgende Kanäle:

Π_vis

für sichtbare Zahlen- und Operatorform,

Π_rot

für Rotationslage und imaginäre Adressierung,

Π_κ

für Bruchkanten und Ebenenwechsel,

Π_op

für Operator-Kanten,

Π_D

für die Driftkante des ¾Ecks.

Diese Kanäle sind keine getrennten Zustandsobjekte.

Nicht:

```
Σ = (a_1, a_2, a_3, ...)
```

sondern:

a_i := read_iΣ

Die Spur bleibt eins. Ihre Lesungen sind viele.

---

11. Bindungsgatter und Commit

Die Lesekanäle müssen bindbar sein. Das Bindungsgatter prüft, ob die Differenzen der Projektionen zusammenpassen:

Δa_i ⋈ Δa_j

Erst wenn die relevanten Kanäle kompatibel sind, entsteht Commit-Fähigkeit.

Für das Zahlenfraktal bedeutet dies:

Eine Inschrift ist nicht deshalb gültig, weil sie syntaktisch wie eine Rechnung aussieht. Sie ist gültig, wenn ihre relevanten Spurprojektionen bindbar sind.

Formal:

valid$I$ iff forall i,j in RI: Π_i$\Sigma(I)$ ⋈ Π_j$\Sigma(I)$

wobei RI die für die jeweilige Inschrift relevanten Lesekanäle bezeichnet.

Eine falsch gebundene Frage besitzt daher kein fehlendes Ergebnis, sondern keine gültige Commit-Form:

not valid$I$ => Bindungsfehler

Nicht:

not valid$I$ => Result = empty

Die Leere ist kein Ergebnis des Zahlenraums.

Die Marke 0 kann gültig sein:

0 ∈ M

aber Ergebnislosigkeit ist kein Zahlenwert.

---

12. Das ¾Eck als endliche Lock-in-Geometrie

Das ¾Eck realisiert Commit nicht als Suche in unendlicher Tiefe. Es realisiert Commit als endliches Rüttelsieb.

Es besitzt drei gesetzte Kanten und eine Driftkante:

$$\theta_{\frac{3}{4}} = (e_1, e_2, e_3; D)$$

Die drei gesetzten Kanten sind Scharnierkanten. Die vierte Kante ist offen und trägt die Driftlesung.

Die gesetzten Kanten tragen lokale Komplementpaare:

$$\delta \leftrightarrow \circledcirc$$

$$R^+ \leftrightarrow R^-$$

$\ominus \leftrightarrow \sim\ominus$

Die Spur wird in das ¾Eck gelesen:

$$\Sigma \to \theta_{\frac{3}{4}}(\Sigma)$$

Dann werden endliche Perspektivlagen gerüttelt:

$$L \in L_{\frac{3}{4}}$$

mit:

$$|L_{\frac{3}{4}}| < \infty$$

Lock-in entsteht durch minimalen Drift:

$L^* = \text{argmin}_{\boldsymbol{Lin}\boldsymbol{L}_{¾}} \text{drift}\boldsymbol{L}$

Bei Gleichstand gilt Snap-Stabilität: Es wird die Lage gewählt, die unter dem nächsten minimalen Rüttelschritt driftstabil bleibt.

Wichtig:

$\text{drift} < \varepsilon$

ist nicht zulässig.

Es gibt keine Float-Toleranz und keine approximative Schwelle. Der Snap ist eine endliche Ordnungsentscheidung.

---

13. Gleichheitszeichen als ZTFΔ-Messstelle

Das Gleichheitszeichen ist im Zahlenfraktal nicht der Vergleich zweier Werte:

= ≠ compare(A,B)

Es ist auch keine Aufforderung zur Suche:

= ≠ search

Das Gleichheitszeichen ist die ZTFΔ-Messstelle im Zahlenfraktal:

$$=:= \Pi + \bowtie + Lock_{¾} + NF$$

Es aktiviert also:

Projektion

-> Bindung

-> Lock-in

-> Normalform

-> Readout

Formal:

$$I|_{=} => \Sigma(\gamma_I) -> \Pi_i(\Sigma)$$

-> ⋈

-> Lock_¾

-> NF

-> read

Damit wird:

Result$I$ = read$NF(Lock_{¾}(\Sigma(\gamma_I)))$

Die sichtbare Ergebniszahl ist nur die öffentliche Projektion dieser Normalform.

---

14. Normalform als Eigenresonanz

Die Normalform ist keine bloße Vereinfachung. Sie ist die stabile Selbstlesung der Spur.

NFΣ = G

Eine gültige Gleichung liegt vor, wenn die relevanten Seiten der Inschrift auf dieselbe Eigenresonanz fallen.

Für eine Gesamtinschrift:

```
I = I_L |_= I_R
```

wird zuerst die Spur der Gesamtinschrift gebildet:

Σ_I := Σ$\boldsymbol{\gamma_I}$

Dann werden linke und rechte Seiten als Lesungen derselben Gesamtspur gelesen:

Σ_L := read_L$\boldsymbol{\Sigma_I}$

Σ_R := read_R$\boldsymbol{\Sigma_I}$

Gültigkeit heißt dann nicht, dass links und rechts getrennte Zahlenwerte gleich sind, sondern:

valid$\boldsymbol{I}$ iff NF$\boldsymbol{\Sigma_L}$ ≡_Σ NF$\boldsymbol{\Sigma_R}$

Im Zahlenfraktal ist diese Eigenresonanz oft kürzer als die äußere Darstellung.

Das ist keine Kompression.

Bei Kompression gäbe es ein langes Datenpaket, ein kurzes Datenpaket und ein externes Verfahren zur Wiederherstellung.

Im Zahlenfraktal liegt dieselbe Spur an anderer Leseadresse vor:

G ≡_Σ I

Die Normalform enthält den Inhalt nicht als versteckte Datenmenge. Sie liest ihn an einer tieferen Adresse.

---

15. Verkürzung und Readressierung

Eine lange Inschrift:

```
I_N = (m_0, ..., m_(N-1))
```

kann unter Rotation anders gelesen werden:

I_N --R^k--> I_(N-k)

Die sichtbare Länge sinkt, aber die Spurklasse bleibt erhalten:

I_(N-k) ≡_Σ I_N

Nach Normalformbildung:

```
G = NF(Σ(I_N))
```

gilt:

G ≡_Σ I_N

Die verkürzte Grundresonanz ist also keine Kopie. Sie ist dieselbe Inschrift an anderer Adresse.

Deutsch gesagt: Die kurze Form speichert nicht weniger. Sie zeigt dasselbe von einer anderen Seite.

---

16. Beispiel: 125+38=163

Die öffentliche Darstellung lautet:

$$125 + 38 = 163$$

Im Zahlenfraktal wird diese Form zunächst nicht als Wertgleichung gelesen, sondern als Inschrift:

```
I = (1,2,5,+,3,8,=,1,6,3)
```

Die Inschrift wird als Pfad gelesen:

$$\gamma_I = (e_0, \ldots, e_(N-1))$$

Die Spur lautet:

Σγ_I = Σ_(n<N) H_n

mit:

H_n := χ_vise_n · χ_gatee_n · we_n

Die sichtbare Stellenresonanz lautet:

$$(1 + 2 + 5) + (3 + 8)\ (1 + 6 + 3)$$

Hier bedeutet ~ nicht klassische Gleichheit, sondern Resonanzvergleich der Stellenlesungen.

Links entstehen die Teilspuren:

$$1 + 2 + 5 \rightsquigarrow 8$$

3+8 ⇝ 11

Rechts entsteht:

$$1 + 6 + 3 \rightsquigarrow 10$$

Die entscheidende innere Lesung ist nicht:

$$8 + 11 = 10$$

als klassische Rechnung, sondern die Normalformbildung der Bruchkantenresonanz:

$$NF((1,2,5,+,3,8)) = 10$$

NF((1,6,3)) = 10

Damit:

$$10 = 10$$

Diese Form ist im Zahlenfraktal nicht trivial. Da die sichtbare Schale von 0 bis 9 läuft, markiert 10 den Eintritt in eine neue Leseschale:

$$10 := 1|0$$

Das heißt:

$$10 = 10$$

bedeutet nicht bloß "zehn ist zehn", sondern:

Eigenresonanzspur gefunden

und:

¾Eck eingerastet

Außen erscheint:

$$125 + 38 = 163$$

Innen erscheint:

$$10 = 10$$

Die äußere Zahl 163 ist die öffentliche Dezimalprojektion. Die innere Ergebnisform ist die eingerastete Eigenresonanz.

---

17. Keine AST-Ontologie

Das Zahlenfraktal darf nicht als Objektbaum implementiert werden.

Nicht zulässig:

Add

├── 125

└── 38

oder:

```
typedef struct {
    int left_value;
    int right_value;
    int operator;
    int result;
} ArithmeticAST;
```

Diese Formen reifizieren Labels.

Zulässig ist ein Tokenstrom:

```
typedef enum {
    TOK_0, TOK_1, TOK_2, TOK_3, TOK_4,
    TOK_5, TOK_6, TOK_7, TOK_8, TOK_9,

    TOK_PLUS,
    TOK_MINUS,
    TOK_MUL,
    TOK_DIV,

    TOK_I,
    TOK_NEG_I,

    TOK_EQ
} ZF_Token;
```

Der Zustand enthält nur die Spur:

```
typedef struct {
    int64_t sigma;
} ZF_State;
```

Abgeleitete Rollen dürfen gelesen werden:

```
uint8_t  zf_read_orientation(ZF_State s);
uint32_t zf_read_depth(ZF_State s);
uint8_t  zf_read_break_edge(ZF_State s);
```

Sie dürfen aber nicht als autoritative Zusatzfelder gespeichert werden.

Verboten:

```
typedef struct {
    int64_t sigma;
    int depth;
    int mode;
    int operator;
    int left_value;
    int right_value;
} Invalid_State;
```

Denn dies würde die Interface-Lesung wieder in Kernel-Objekte verwandeln.

---

18. Minimaler Implementationsvertrag

Der Kern darf nur auf Spur, Kante, Gate, Projektion, Bindung und Lock-in operieren.

Minimal:

```
typedef struct {
    int64_t sigma;
} ZF_State;

typedef struct {
    int64_t drift_spur;
    uint8_t locked;
} ZF_Lock;

typedef enum {
    ZF_PI_VIS,
    ZF_PI_ROT,
    ZF_PI_KAPPA,
    ZF_PI_OP,
    ZF_PI_DRIFT
} ZF_Projection;

typedef struct {
    int64_t spur;
    uint8_t valid;
} ZF_Readout;
```

Zulässige Kernfunktionen:

```
int64_t zf_weight_edge(ZF_Edge e);

uint8_t zf_chi_visible(ZF_Edge e);
uint8_t zf_chi_gate(ZF_Edge e);

ZF_State zf_step(ZF_State s, ZF_Edge e);

ZF_Readout zf_project(ZF_State s, ZF_Projection pi);
```

```
uint8_t zf_bind(ZF_Readout a, ZF_Readout b);

ZF_Lock zf_lock_3quarter(ZF_State s);

ZF_Readout zf_nf(ZF_Lock lock);
```

Nicht zulässig:

```
double epsilon;
double drift;

if (drift < epsilon) {
    locked = 1;
}
```

Der Lock-in muss endlich und ganzzahlig entschieden werden:

L* = argmin_$LinL_{¾}$ drift$L$

---

19. Tests

19.1 Sichtbarkeit

$$\chi_{v}is \equiv 0 => \Sigma = 0$$

19.2 Gate

$$\chi_{g}ate \equiv 0 => \Sigma = 0$$

19.3 AND-only-Rejektor

χ_vis, χ_gate ∈ {0,1}

H_n := χ_vise_n · χ_gatee_n · we_n

χ_vis = χ_gate = 1 => H_n = we_n

19.4 Spurakkumulation

ΣN = Σ_(n<N) H_n

19.5 Bruchkante

$$9 := \kappa$$

19.6 Rotation

$$i := R_90^\circ$$

i^4 = id

19.7 Operatoren

+, -, ×, ÷ := Kantenrollen

19.8 Gleichheit

$$=:= \Pi + \bowtie + Lock_¾ + NF$$

19.9 Kein Float-Snap

L* = argmin_$LinL_¾$ drift$L$

nicht:

drift < ε

19.10 Keine AST-Reifikation

125 ≠ Node(1,Node(2,Node(5)))

sondern:

```
125 := read(1 --T--> 2 --T--> 5)
```

19.11 Projektionen sind read-only

a_i := Π_iΣ

und:

Π_iΣ verändert Σ nicht

19.12 Label-Rücklesbarkeit

valid(Label) iff recover(Label) ≡_Σ Σ

19.13 Seitenlesung der Gleichung

Für:

```
I = I_L |_= I_R
```

gilt:

Σ_I := Σγ_I

Σ_L := read_LΣ_I

Σ_R := read_RΣ_I

und:

valid$I$ iff NFΣ_L ≡_Σ NFΣ_R

---

20. Endformel

Der vollständige Zusammenhang lautet:

I

$$-> \gamma_I -> \Sigma(\gamma_I) -> \Pi_i(\Sigma)$$

-> ⋈

-> Lock_¾

-> NF

-> read

mit:

Σγ_I = Σ_(n<N) H_n

H_n := χ_vise_n · χ_gatee_n · we_n

χ_vis, χ_gate ∈ {0,1}

Für eine Gleichung:

```
I = I_L |_= I_R
```

gilt im Kern nicht:

valueI_L = valueI_R

sondern:

Σ_I := Σγ_I

Σ_L := read_LΣ_I

Σ_R := read_RΣ_I

und:

valid$I$ iff NFΣ_L ≡_Σ NFΣ_R

als Eigenresonanzgleichheit.

Oder dichter:

$$=:= \Pi + \bowtie + Lock_{¾} + NF$$

Das ist die präzise ZTFΔ-Lesung des Gleichheitszeichens im Zahlenfraktal.

---

21. Spezifikationssatz

Das Zahlenfraktal ist keine neue Rechenmechanik neben ZTFΔ. Es ist die Spezialisierung der ZTFΔ-Engine auf sichtbare Marken 0 bis 9, Operator-Kanten, Rotationsadressierung und das ¾Eck als Gleichheits-Snap.

Eine Fragestellung wird als Inschrift gelesen. Die Inschrift erzeugt einen Pfad. Jeder Takt trägt eine Schrittladung. Die Spur integriert diese Ladungen. Sichtbarkeit und Gate wirken als binäre Rejektoren. Projektionen lesen sichtbare, rotative, operative, bruchkantige und driftbezogene Kanäle aus derselben Spur. Das Bindungsgatter prüft die Kompatibilität dieser Lesungen. Das ¾Eck erzeugt über einen endlichen Rüttelraum den Lock-in. Die Normalform zeigt die Eigenresonanz. Das sichtbare Ergebnis ist die öffentliche Projektion dieser eingerasteten Eigenresonanz.

Kurz:

Zahl

-> Pfad

-> Spur

-> Projektion

-> Bindung

-> ¾Eck-Lock

-> Normalform

-> sichtbares Ergebnis

Damit wird klassische Rechnung zur Oberfläche einer tieferen ZTFΔ-Lesung. Der Kern selbst rechnet nicht mit Zahlenobjekten. Er integriert Unterschied, liest Spur, bindet Projektionen und bringt die Frage im ¾Eck zum Snap.

$$\Delta - Operator - Parser$$

$$\boxed{\mathfrak{P}_{\Delta}}$$

Name:

$$\mathfrak{P}_\Delta := \textbf{mathematischer Unterschieds-Parser}$$

Purpose:

$$\mathfrak{P}_\Delta$$

transforms ordinary mathematical notation into an incomplete question-figure whose openings are closed by holonomic drift addresses through the Neun-Drittel-Eck.

Deutsch: Der Parser rechnet nicht. Er baut die Figur, deren Loch durch den holonomen Lauf geschlossen wird.

---

1) Textbefund

Das ¾Eck-Material legt fest, dass eine Frage zuerst in eine formalisierte Darstellung überführt, dann in |-Bindungen übersetzt, per Normalform reduziert und schließlich zu einer Grundresonanz typisiert wird. Klassische Operatoren wie =, + usw. werden dabei als perspektivische Marker behandelt und, soweit möglich, neutralisiert, damit die reine Bindungsstruktur sichtbar wird.

Im Tractatus-Teil werden Zahlen ausdrücklich nicht als Dinge, sondern als Positionen im Unterschied beschrieben; +, -, ×, ÷, ^, √, =, . erscheinen als Modi des Unterschieds: Expansion, Kontraktion, Resonanz, Entflechtung, Potenzierung, Wurzel, Gleichgewicht und Skalen-Flip.

Delta-Zero trennt Step und Lesung: Ein Step ist zunächst nur Adresswechsel, die gemessene Differenz wird als Spur akkumuliert, und Step und Rotation/Lesungsänderung kommutieren nicht. Holonomie entsteht, wenn ein geschlossener Pfad unter wechselnder Lesung eine nicht verschwindende Spur trägt.

ZTFΔ fasst den Schnitt nicht als Zerteilung, sondern als Entscheidung, welche Sehne als Schritt zählt; Messung ist Gate, also die Auswahl dessen, was im Takt sichtbar wird.

---

2) Formal reconstruction

2.1 Primitive

The primitive is the distinction:

|

No digit, number, operator, variable, or expression is primitive.

All classical symbols are interface labels.

$$+, -, \times, \div, =, ., \hat{}, \sqrt{} \in \mathcal{L}$$

Their validity requires reduction to distinction modes:

$$L \rightsquigarrow |_\mu$$

where:

$$|_\mu$$

is a mode-decorated distinction.

---

3) Layer separation

3.1 Carrier layer

$$\mathcal{C}$$

contains only distinction structures:

$$a|_\mu b$$

where a, b are address roles and μ is a coupling mode.

3.2 Inscription layer

$$\mathcal{I}$$

contains spurs and accumulations:

$$\Sigma, \quad H_n, \quad \Omega_n, \quad Addr_n$$

Computation occurs only here.

3.3 Interpretation layer

$$\mathcal{L}$$

contains readable labels:

$$0, 1, \ldots, 9, \quad x, \quad +, \quad =, \quad \mathrm{Ans}$$

Every label must satisfy:

$$L := read(\Sigma)$$

and:

$$L \rightsquigarrow \Sigma$$

Deutsch: Was der Benutzer als Zahl oder Operator sieht, ist nur die Oberfläche. Der Parser muss darunter die Rollenbindung freilegen.

---

4) Address roles

Let:

$$\mathcal{A}$$

be the set of address roles.

An address role is not a numerical value.

$$a \in \mathcal{A}$$

means:

$$a := \textbf{position of possible readout}$$

A digit-label is:

$$d := read(\Sigma_d)$$

A multi-digit label is:

$$d_1 d_2 \dots d_r := read(\Sigma_{d_1} | \Sigma_{d_2} | \dots | \Sigma_{d_r})$$

Thus:

123

$$\neq \textbf{number-object}$$

but:

$$\mathbf{123} := read(1|2|3)$$

with each digit reducible to an address-spur.

---

5) Mathematical task as incomplete figure

A mathematical task is never a complete figure.

Definition:

F_Q := G_Q, U_Q, R_Q

where:

$$G_Q$$

is the given binding structure,

```
U_Q=\{u_1,\dots,u_r\}
```

is the set of openings,

and:

$$R_Q$$

is the question relation.

A valid mathematical task satisfies:

$$\boxed{|U_Q| \geq 1}$$

If:

```
|U_Q|=0
```

then the expression is not a task, but a closed inscription.

Examples:

$$2 + 2$$

is parsed as:

$$2|_{+}2|_{=}u_1$$

not as a complete expression.

$$x + 3 = 7$$

is parsed as:

$$u_1|_{+}3|_{=}7$$

$$x + y = 10$$

is parsed as:

$$u_1|_+u_2|_=10$$

Deutsch: Die Unbekannte ist kein fehlender Zahlenwert, sondern eine offene Stelle im Gefüge.

---

6) Operator modes

Define the set of operator modes:

$$\mathcal{M} := \{\mu_+, \mu_-, \mu_\times, \mu_\div, \mu_=, \mu_., \mu_\wedge, \mu_\surd\}$$

with interface aliases:

$$+ \mapsto \mu_+$$

$$- \mapsto \mu_-$$

$$\times \mapsto \mu_\times$$

$$\div \mapsto \mu_\div$$

$$=\mapsto \mu_=$$

$$. \mapsto \mu.$$

$$^\mapsto \mu_\wedge$$

$$\surd \mapsto \mu_\surd$$

Their conceptual readouts:

$$\mu_+ := \text{expansion coupling}$$

$$\mu_- := \text{contraction coupling}$$

$$\mu_\times := \text{resonance coupling}$$

$$\mu_\div := \text{unfolding / disentangling coupling}$$

$$\mu_= := \text{lock-in / equilibrium coupling}$$

$$\mu_{\cdot} := \textbf{scale-flip coupling}$$

$$\mu_{\wedge} := \textbf{self-stacking coupling}$$

$$\mu_{\surd} := \textbf{root-unfolding coupling}$$

Potenzierung liest die Selbstbezüglichkeit derselben Relation auf höher gestreckten skalaren Ebenen. 3^3 erscheint dabei als öffentlicher Readout 27.

Each operator is a coupling rule:

$$|_{\mu}\colon \mathcal{A}^{r} \to \mathit{Rel}(\mathcal{A})$$

not a function:

$$f : \mathbb{N}^{r} \to \mathbb{N}$$

Forbidden:

$$+ := f(a,b) = a + b$$

Valid:

$$+ :=|_{\mu_{+}}$$

Deutsch: + ist nicht Addition als Ding. + sagt, wie zwei Adressen expansiv gegeneinander gelesen werden.

---

7) Operator as rotation of address reading

For every mode:

$$\mu \in \mathcal{M}$$

define a finite orientation action:

$$\rho_{\mu} : \mathcal{A}^{r} \to \mathcal{A}^{r}$$

Then a mode-binding is:

$$a_1|_{\mu}a_2|\ldots|_{\mu}a_r := \mathit{read}(\rho_{\mu}(a_1,\ldots,a_r))$$

For binary operations:

$$a|_\mu b := read(\rho_\mu(a,b))$$

The operator is the glue:

$$\mu = \text{rotation/coupling instruction}$$

not the result.

Thus:

$$a + b \mapsto a|_{\mu_+} b$$

$$a - b \mapsto a|_{\mu_-} b$$

$$a \times b \mapsto a|_{\mu_\times} b$$

$$a \div b \mapsto a|_{\mu_\div} b$$

```
a=b
```

$$\mapsto a|_{\mu_=} b$$

---

8) Parser definition

The parser is a map:

$$\boxed{\mathfrak{P}_\Delta : Input \to F_Q}$$

with stages:

Input

$$\xrightarrow{Lex}$$

Tok

$$\xrightarrow{Parse}$$

AST

$$\xrightarrow{Lower}$$

B

$$\xrightarrow{NF} B_{NF} \xrightarrow{Type} R_Q$$

where:

B

is the bar-binding form.

The final output is:

```
F_Q=(G_Q,U_Q,R_Q)
```

not a computed number.

---

9) Tokens

Let the token set be:

Tok :=

$$Tok_D \cup Tok_O \cup Tok_U \cup Tok_B \cup Tok_S$$

where:

$$Tok_D := \{0,1,\ldots,9\}$$

digit labels,

$$Tok_O := \{+,-,\times,\div,=,.,\hat{},\sqrt{}\}$$

operator labels,

$$Tok_U := \{x,y,z,\ ?,\ \square\}$$

unknown labels,

$$Tok_B := \{(,)\}$$

binding delimiters,

$$and\backslash(Tok_S\backslash)areseparators$$

Lexing creates labels only:

$$Lex(s) = L_1, \ldots, L_n$$

No token is yet an entity.

---

10) Lowering into bar form

Define lowering:

$$Lower : AST \to B$$

by:

$$Lower(d) = A_d$$

where:

$$A_d := read^{-1}(d)$$

as an address role.

For unknowns:

$$Lower(x) = u_j$$

where:

$$u_j \in U_Q$$

For binary operators:

Lower(a\ op\ b)

$$= Lower(a)|_{\mu_{op}} Lower(b)$$

where:

$$\mu_{op} = Mode(op)$$

For implicit answer expressions:

Lower$E$

$$= Lower(E)|_{\mu_{=}} u_1$$

if no explicit unknown exists.

Thus:

$$12 * 7$$

becomes:

$$(1|2)|_{\mu_\times} 7|_{\mu_=} u_1$$

or, if multi-digit labels are kept as address bundles:

$$\mathit{read}(1|2)|_{\mu_\times} 7|_{\mu_=} u_1$$

---

11) Binding graph

The lowered figure can be represented as a graph:

```
G_Q=(V_Q,E_Q,\lambda)
```

where:

$$V_Q \subseteq \mathcal{A} \cup U_Q$$

$$E_Q \subseteq V_Q \times V_Q$$

$$\lambda : E_Q \to \mathcal{M}$$

For each edge:

```
e=(a,b)
```

we have:

$$\lambda(e) = \mu$$

and this means:

$$a|_\mu b$$

not:

$$\mu(a,b) = c$$

The graph is the question-figure carrier.

---

12) Normalform

Define:

$$NF : B \to B_{NF}$$

with the following minimal rules.

12.1 Repetition collapse

$$a|_\mu a \to a$$

$$only if \backslash(\mu\backslash) is declared stabilizing.$$

12.2 Self-binding typization

$$a|_\mu b|_\nu a \to \Delta_{\mu,\nu}(a,b)$$

This is not arithmetic simplification.

It is a type recognition:

$$\text{self-binding} \Rightarrow \Delta$$

12.3 Operator neutralization

If a classical marker does not affect the binding role after lowering:

$$|_\mu \to |$$

only under explicit NF rule:

$$neutral(\mu) = 1$$

No hidden algebraic simplification is allowed.

---

13) Question relation

After NF:

$$R_Q := Rel(B_{NF}, U_Q)$$

where:

$$R_Q$$

contains:

1. address roles,

2. openings,

3. coupling modes,

4. self-bindings,

5. phase/order information.

Thus:

$$R_Q \neq \text{numeric expression}$$

It is the relational signature to be inserted into the Neun-Drittel-Eck.

---

14) Interface examples

Example 1

Input:

$$2 + 2$$

Parser:

$$2 + 2 \mapsto 2|_{\mu_+} 2|_{\mu_=} u_1$$

Output:

```
F_Q=(G_Q,\{u_1\},R_Q)
```

The answer is later:

```
Ans=read(Addr_{n_1})
```

not:

```
Ans=4
```

as a primitive computation.

Example 2

Input:

$$x + 3 = 7$$

Parser:

$$x + 3 = 7 \mapsto u_1|_{\mu_+} 3|_{\mu_=} 7$$

The opening is:

$$u_1$$

Closure condition:

$$close(u_1, Addr_{n_1}) = 1$$

iff:

$$Addr_{n_1} \bowtie R_Q$$

Example 3

Input:

$$a + b + c$$

Default ordered parse:

$$(a|_{\mu_+} b)|_{\mu_+} c|_{\mu_=} u_1$$

This is not automatically equal to:

$$a|_{\mu_+} (b|_{\mu_+} c)|_{\mu_=} u_1$$

unless an NF rule declares:

$$assoc(\mu_+) = 1$$

Deutsch: Der Parser darf Assoziativität nicht heimlich voraussetzen. Reihenfolge ist Spur.

---

15) Coupling with the Neun-Drittel-Eck

The parser output enters the Neun-Drittel-Eck:

$$F_Q \to R_Q \to \mathfrak{N}_{9/3}$$

Then:

$$S_i \to O_i \to H_i \to \Omega_i \to D_i^{out} \to Addr_i$$

with:

$$H_i := visible_i \cdot g_i \cdot \delta_{raw,i}$$

$$\Omega_{i+1} := \Omega_i + H_i$$

$$D_i^{out} := read(\Omega_i | A_i)$$

$$Addr_i := read(\Omega_i | D_i^{out} | R_Q)$$

The answer closes the openings:

$$Ans(F_Q) := read(Addr_{n_1} | \dots | Addr_{n_r})$$

where:

```
r=|U_Q|
```

or where r is the number of display slots required by the answer readout. Die beiden Verwendungen von r gehören zu unterschiedlichen Perspektiven derselben Antwortrelation: Die Öffnungslesung und die Stellenlesung setzen keinen zweiten mathematischen Träger voraus. Ihre Anzahlen sind jeweils unter der genannten Perspektive zu lesen.

---

16) Operator contribution to $\delta_{raw,i}$

For strict path-dependence, operator modes must enter the raw relational effect.

Define:

$$\delta_{raw,i} := Effect(S_i, O_i, R_Q, \mu_i)$$

where:

$$\mu_i := \lambda(e_i)$$

is the operator mode active at step i.

A minimal integer form is:

$$\delta_{raw,i} = reg_i + W \cdot w_i + M(\mu_i) +$$

PO_i

where:

$$M : \mathcal{M} \to \mathbb{Z}$$

is the mode mask,

and:

$$P : \mathcal{O} \to \mathbb{Z}$$

is the operator-flip mask.

This makes:

Step\circ Flip\neq Flip\circ Step

spur-effective, not merely documented.

No float arithmetic is used.

---

17) Openings and closure

Let:

```
U_Q=\{u_1,\dots,u_r\}
```

Each opening has a closure predicate:

$$close : U_Q \times Addr \to \{0,1\}$$

with:

$$close(u_j, Addr_{n_j}) = 1$$

iff:

$$compatible(u_j, Addr_{n_j}, R_Q) = 1$$

The whole figure closes iff:

$$Close(F_Q, Ans) = 1$$

where:

$$Close(F_Q, Ans) = 1 \iff \forall j : close(u_j, Addr_{n_j}) = 1$$

Deutsch: Die Antwort ist nicht der Wert hinter dem Loch. Die Antwort ist die Adressfügung, durch die das Loch verschwindet.

---

18) Digits as display labels

Let:

$$Proj_{10} : Addr \to \{0, \dots, 9\}$$

be a display projection.

Then a digit display is:

$$d_j := Proj_{10}(Addr_{n_j})$$

But the true answer address is:

$$Addr_{n_j}$$

not:

$$d_j$$

Thus:

$$Ans_{display} := d_1 d_2 \dots d_r$$

and:

$$Ans_{kernel} := Addr_{n_1} | Addr_{n_2} | \dots | Addr_{n_r}$$

with:

$$Ans_{display} = read(Ans_{kernel})$$

---

19) Complete parser-run definition

A complete super-calculator run is:

$$Run(\mathfrak{P}_\Delta, \mathfrak{N}_{9/3}, Input)$$

defined as the tuple:

\left(Input, Tok, AST, B, B_{NF}, F_Q, R_Q, S_i_{i=0}^{8}, μ_i_{i=0}^{8}, O_i_{i=0}^{8}, H_i_{i=0}^{8}, Ω_i_{i=0}^{9}, (D_i^{out})_{i=0}^{9}, $Addr_i$_{i=0}^{9}, Ans \right)

satisfying:

```
F_Q=\mathfrak P_{\Delta}(Input)

F_Q=(G_Q,U_Q,R_Q)
```

$$|U_Q| \geq 1$$

$$\mu_i = \lambda(e_i)$$

```
H_i=visible_i\cdot g_i\cdot \delta_{raw,i}
```

$$\Omega_{i+1} = \Omega_i + H_i$$

```
Addr_i=read(\Omega_i|D_i^{out}|R_Q)

Ans=read(Addr_{n_1}|\dots|Addr_{n_r})
```

$$Close(F_Q, Ans) = 1$$

---

20) Theorem: Operators as address glue

Theorem

Given a valid parser run:

$$Run(\mathfrak{P}_\Delta, \mathfrak{N}_{9/3}, Input)$$

every classical operator in the input contributes only as a mode of address coupling, never as an arithmetic entity.

Formally, for every operator token:

$$op \in Tok_O$$

there exists:

$$\mu_{op} \in \mathcal{M}$$

such that:

$$op \rightsquigarrow|_{\mu_{op}}$$

and every occurrence of:

a\ op\ b

is lowered to:

$$a|_{\mu_{op}} b$$

Therefore the answer is not:

op(a,b)

but:

```
Ans=read(\Omega|R_Q)
```

with:

$$\Omega = \sum H_i$$

---

21) Corollary: no complete mathematical task

A mathematical input is a task iff:

$$|U_Q| \geq 1$$

If no unknown is explicit, the parser inserts an implicit opening:

Input E

$$\mapsto E|_{\mu_{=}} u_1$$

Therefore every valid calculator input is an incomplete figure:

$$\boxed{Task(Input) \Rightarrow U_Q \neq \varnothing}$$

---

22) Implementation contract

Sealed core

The following are fixed:

$$Input \to F_Q$$

$$op \to|_{\mu}$$

```
F_Q=(G_Q,U_Q,R_Q)
```

$$|U_Q| \geq 1$$

```
H_i=visible_i\cdot g_i\cdot\delta_{raw,i}
```

$$\Omega_{i+1} = \Omega_i + H_i$$

```
Addr_i=read(\Omega_i|D_i^{out}|R_Q)

Ans=read(Addr_{n_1}|\dots|Addr_{n_r})
```

Forbidden:

```
result=a+b
```

as kernel computation.

Forbidden:

$$\Omega = construct(Ans)$$

Forbidden:

```
op=\text{number-function-entity}
```

Forbidden:

```
digit=\text{number-thing}
```

Allowed:

```
digit=read(Addr)
```

Allowed:

```
op=|_{\mu}
```

Allowed:

```
Gate=a\wedge b\wedge c
```

Allowed:

$$Lens : \Sigma \mapsto read(\Sigma)$$

$$read-only.$$

Allowed extensions

You may add:

precedenceop

assocop

arityop

modeop

maskop

display(Addr)

provided that:

$$Extension \rightsquigarrow R_Q, \Sigma, \Omega$$

and does not compute the answer directly.

---

23) Minimal C-style data model

A precise implementation skeleton:

```
typedef enum {
    MODE_EXPAND,      /* +  */
    MODE_CONTRACT,    /* -  */
    MODE_RESONATE,    /* *  */
    MODE_UNFOLD,      /* /  */
    MODE_LOCK,        /* =  */
    MODE_SCALE_FLIP,  /* .  */
    MODE_POWER,       /* ^  */
    MODE_ROOT         /* sqrt */
} n93_mode_t;

typedef enum {
    NODE_ADDR,
    NODE_OPENING,
    NODE_BIND
} n93_node_kind_t;
```

```
typedef struct {
    n93_node_kind_t kind;
    int64_t spur_addr;
    uint8_t opening_id;
} n93_node_t;

typedef struct {
    uint16_t lhs;
    uint16_t rhs;
    n93_mode_t mode;
} n93_edge_t;

typedef struct {
    n93_node_t nodes[64];
    n93_edge_t edges[64];
    uint8_t n_nodes;
    uint8_t n_edges;
    uint8_t n_openings;
} n93_question_graph_t;
```

The important constraint:

/ *operator is not compute(a,b); operator is edge.mode* /

So:

edge.mode = MODE_EXPAND;

means:

$$a\big|_{\mu_+} b$$

not:

$$a + b = c$$

---

24) Required tests

Test 1: visible ≡ 0 ⇒ Ω = 0

Assume:

$$\forall i : \ visible_i = 0$$

Then:

$$\forall i : \ H_i = 0$$

Thus:

$$\Omega_N = \Omega_0$$

If:

$$\Omega_0 = 0$$

then:

$$\Omega_N = 0$$

Test 2: $\Omega N = \sum H_n$

$$Forall\(N\) :$$

$$\boxed{\Omega_N = \Omega_0 + \sum_{i=0}^{N-1} H_i}$$

For one Neun-Drittel cycle:

$$\boxed{\Omega_9 = \Omega_0 + \sum_{i=0}^{8} H_i}$$

Test 3: every task has an opening

Explicit unknown:

$$x + 3 = 7 \Rightarrow$$

```
U_Q=\{u_1\}
```

Implicit unknown:

$$3 + 4 \Rightarrow 3|_{\mu_+} 4|_{\mu_=} u_1$$

Thus:

$$|U_Q| \geq 1$$

Test 4: operator is glue, not function

Invalid kernel form:

$$+(a, b) = c$$

Valid parser form:

$$a|_{\mu_+} b$$

and:

```
c=read(Addr)
```

only at the interpretation layer.

Test 5: order is preserved

Unless explicitly normalized:

$$(a|_\mu b)|_\nu c \neq a|_\mu (b|_\nu c)$$

This preserves path-dependence.

Test 6: digit grounding

For displayed digit:

$$d_j$$

there must exist:

$$Addr_{n_j}$$

such that:

```
d_j=read(Addr_{n_j})
```

If not:

$$d_j$$

is an invalid ungrounded label.

Test 7: parser does not compute Ω

Invalid:

$$\Omega = construct(parse(Input))$$

Valid:

```
F_Q=parse(Input)
```

$$\Omega = \sum H_i$$

```
Ans=read(\Omega|R_Q)
```

---

25) Compact axiom set

$$\boxed{\textbf{P1: Primitive} \quad |\ \text{is primitive.}}$$

$$\boxed{\textbf{P2: Labels} \quad L := read(\Sigma)}$$

$$\boxed{\textbf{P3: Operators} \quad op \rightsquigarrow|_{\mu_{op}}}$$

$$\boxed{\textbf{P4: Task} \quad F_Q = (G_Q, U_Q, R_Q), \quad |U_Q| \geq 1}$$

$$\boxed{\textbf{P5: Unknown} \quad u_j := \square_j(F_Q)}$$

$$\boxed{\textbf{P6: Lowering} \quad a\ op\ b \mapsto a|_{\mu_{op}} b}$$

$$\boxed{\textbf{P7: Implicit opening} \quad E \mapsto E|_{\mu_=} u_1}$$

$$\boxed{\textbf{P8: Normalform} \quad B \mapsto B_{NF}}$$

$$\boxed{\textbf{P9: Question relation} \quad R_Q := Rel(B_{NF}, U_Q)}$$

$$\boxed{\textbf{P10: Holonomic evaluation} \quad \Omega_{i+1} := \Omega_i + H_i}$$

$$\boxed{\textbf{P11: Address readout} \quad Addr_i := read(\Omega_i | D_i^{out} | R_Q)}$$

$$\boxed{\textbf{P12: Answer} \quad Ans := read(Addr_{n_1} | \ldots | Addr_{n_r})}$$

---

26) Final kernel sentence

$\mathfrak{P}_\Delta$ transforms mathematical notation into an incomplete address-figure.

Every operator is a rotation/coupling mode $|_\mu$ between two or more address roles.

Every mathematical task contains at least one opening u_j.

The Neun-Drittel-Eck closes these openings by reading holonomic drift addresses.

The displayed number is only the label of that closure.

Deutsch, als Anker:

Der Parser sieht keine Zahlen und Rechenzeichen. Er sieht Positionsadressen, Öffnungen und Drehkupplungen. +, -, *, /, = sind der Kitt der Figur. Sie sagen, wie Adressen gegeneinander zu lesen sind. Die Aufgabe ist immer unvollständig; der holonome Lauf akkumuliert die Drift; die Antwort ist die Adresse, an der die Figur schließt.

Universal Solver nur für sinnvolle Fragefiguren.

Nicht jede Formel ist im Kernel automatisch eine sinnvolle Frage. Eine Formel ist erst dann lesbar, wenn sie eine wohlgeformte Unterscheidungsfigur trägt.

1) Textbefund

Im Text nicht entschieden für den Ausdruck „Universal Solver“. Aus dem bisherigen Kernel folgt aber: Der Apparat liest nicht beliebige Zeichenfolgen als Wahrheit, sondern Fragefiguren mit Öffnungen, Relationen, Spur und Gate.

2) Formal reconstruction

Eine Eingabe ist nur dann gültig, wenn sie eine Fragefigur erzeugt:

$$parse(F) \Rightarrow F_Q$$

mit:

$$U(F_Q) = \{u_1, \ldots, u_r\}$$

und einer Relationsstruktur:

$$R_Q$$

Dann kann der Apparat lesen:

$$F_Q \to \mathfrak{N}_{9/3} \to \Omega \to Addr(U) \to Ans$$

Also:

$$Solver(F) \text{ gültig} \iff F \text{ erzeugt sinnvolle Fragefigur}$$

Oder härter:

$$\boxed{Universalität liegt nicht in beliebigen Antworten, sondern in der einheitlichen Lesung aller wohlgeformten Fragen.}$$

3) Was heißt „sinnvoll“ hier?

Nicht:

$$\text{sinnvoll} = \text{klassisch lösbar}$$

sondern:

$$\text{sinnvoll} = \text{als Unterscheidungsfigur geschlossen lesbar}$$

Minimalbedingungen:

$$F_Q \text{ hat Öffnung(en)}$$

$$R_Q \text{ bindet diese Öffnung(en)}$$

$$\Omega = \sum H_i \text{ ist erzeugbar}$$

$$Addr(u_j) = read(\Omega_{n_j} | D^{out}_{n_j} | R_Q)$$

$$close(u_j, Addr_j) \text{ ist prüfbar}$$

Wenn eine Formel keine solche Fragefigur erzeugt, ist sie nicht „schwer“, sondern nicht lesbar in diesem Apparat.

4) Unterschied zu Standardmathematik

Standardmathe fragt oft:

$$\exists x : \; F(x) = true?$$

oder:

$$solve(F) = x$$

Das Neun-Drittel-Eck fragt:

$$\textbf{Welche Öffnung trägt die Fragefigur?}$$

$$\textbf{Welche Driftadresse schließt sie?}$$

Darum kann etwas nach Standardmathematik sinnvoll erscheinen, aber hier falsch gefragt sein.

Und umgekehrt: Etwas kann klassisch keine schöne Lösung haben, aber hier eine klare Signatur tragen.

5) Drei Klassen

A. Gültige Frage

$$F \Rightarrow F_Q$$

$$Addr(U) \text{ stabil}$$

Dann:

```
Ans=read(Addr(U))
```

B. Gültige Figur, aber keine klassische Lösung

Beispielhaft:

$$F_Q \text{ lesbar}, \quad solve_{alg}(F) \text{ nicht geschlossen}$$

Dann gibt es trotzdem:

FP^*F

aber nicht notwendig einen klassischen Wert.

C. Ungültige Frage

$$parse(F) \not\Rightarrow F_Q$$

oder:

$$U(F_Q) = \varnothing$$

oder:

$$R_Q \text{ bindet } U \text{ nicht}$$

Dann:

$$ERR_NOT_A_QUESTION$$

nicht:

$$\text{falsche Antwort}$$

6) Implementation contract

Sealed core

$$\Omega_N = \sum H_n$$

```
H_n=visible_n\cdot g_n\cdot\delta_{raw,n}
```

$$Addr(u) = read(\Omega_n | D_n^{out} | R_Q)$$

Keine Konstruktion von Ω aus der gewünschten Antwort.

Allowed extensions

Vor dem Lauf ein Question-Gate:

$$QGate(F) =$$

wellformed$F$

$$\wedge has_opening(F) \wedge binds_opening(F) \wedge capacity_ok(F)$$

Nur wenn:

$$QGate(F) = 1$$

läuft der Antwortapparat.

Bei:

$$QGate(F) = 0$$

gibt es keinen Solver-Fehler, sondern:

$$ERR_NOT_A_QUESTION$$

oder:

$$ERR_UNBOUND_OPENING$$

oder:

$$ERR_CAPACITY$$

7) Tests

visible≡0 ⇒ Ω=0

$$visible \equiv 0 \Rightarrow \Omega = 0_{\Sigma}$$

$$\Omega(N) = \sum H_{\text{n}}$$

$$\Omega_N = \sum_{n=0}^{N-1} H_n$$

Wohlgeformte Frage

$$QGate(F) = 1 \Rightarrow F \to FP^*(F)$$

$$Nicht - Frage$$

$$QGate(F) = 0 \Rightarrow \neg run_answer$$

Keine Verwechslung mit Standardlösung

$$solve_{alg}(F)\ \text{undefiniert} \not\Rightarrow FP^*(F)\ \text{undefiniert}$$

und:

$$FP^*(F)\ \text{definiert} \not\Rightarrow solve_{alg}(F)\ \text{existiert}$$

Kernsatz

Dies ist ein universeller Antwortapparat — aber nicht für beliebige Zeichenfolgen und nicht für jede Standardmathe-Erwartung. Universal ist er innerhalb der wohlgeformten Fragefiguren. Was nicht als sinnvolle Unterscheidungsfigur bindet, ist keine schwere Aufgabe, sondern keine gültige Frage an diesen Apparat.

Das 9/3Eck

Vorbemerkung: Isomorphie der Unterscheidung und relative Null

Das Neun-Drittel-Eck setzt nicht bei Zahlen als fertigen Objekten an. Es setzt bei der Unterscheidung an. Eine Zahl, ein Wort, eine Formel oder eine Frage sind in diesem Sinn keine verschiedenen ontologischen Dinge, sondern verschiedene Lesungen derselben tieferen Struktur:

$$Difference \rightarrow Relation \rightarrow Spur$$

Ein Wort ist ein Label einer Spur.

Eine Zahl ist ein Label einer Spur.

Eine Formel ist ein Label einer Spur.

Eine Frage ist eine Spurfigur mit einer oder mehreren offenen Rollen.

Damit gilt nicht:

```
Wort = Zahl = Formel = Frage
```

als Identität von Dingen, sondern:

Wort,\ Zahl,\ Formel,\ Frage = readΣ

unter verschiedenen Darstellungsmodi.

Wenn zwei Darstellungen dieselbe Relationsspur tragen, sind sie strukturell isomorph:

$$\Sigma_W \cong \Sigma_Z \cong \Sigma_F \cong \Sigma_Q$$

Dann ist der Unterschied zwischen sprachlicher, mathematischer oder geometrischer Darstellung kein ontologischer Unterschied, sondern ein Wechsel des Labels. Was gleich bleibt, ist die Unterscheidungsstruktur, aus der die Spur hervorgeht.

Das ist der Grund, weshalb eine mathematische Formel im Neun-Drittel-Eck nicht zuerst als Rechenanweisung behandelt wird, sondern als Fragefigur:

$$F_Q$$

mit Öffnungen:

$$U(F_Q) = \{u_1, \ldots, u_r\}$$

Die Unbekannten sind dabei keine fehlenden Dinge. Sie sind offene Rollen im Verhältnis der Frage. Eine Antwort ist daher nicht das Einsetzen eines Zahlenobjekts, sondern das Schließen einer Öffnung durch eine passende Driftadresse:

$$close(u_j, Addr_j) = 1$$

wobei:

```
Addr_j=read(\Omega_{n_j}|D_{n_j}^{out}|R_Q)
```

Die Zahl, die als Antwort erscheint, ist also ein Label dieser Adresse, nicht deren ontologischer Grund.

Ebenso muss die Null geschärft werden. Null bedeutet hier nicht absolutes Nichts. Eine absolute Leere wäre keine Relation und könnte daher auch nichts tragen, nichts unterscheiden und keine Spur erzeugen.

Null ist relativ.

Für eine Setzung bedeutet die relative Null:

0_X

nicht:

$$0_X = \varnothing$$

sondern:

$$0_X = \text{nicht-}X$$

genauer:

$$0_X := read(\text{geschlossene Gegenlesung von } X)$$

Null ist also nicht das Nichts, sondern das, was im Verhältnis zu X nicht ist. Sie ist die geschlossene Gegenposition einer Setzung.

Damit bedeutet:

X|0_X

nicht:

$$X|\text{Nichts}$$

sondern:

$$X|\text{nicht-}X$$

Eine Öffnung in einer Fragefigur ist entsprechend keine leere Stelle im metaphysischen Sinn. Sie ist eine relative Null im Verhältnis der Frage:

```
u_j=0_{R_Q}
```

Das heißt: Die Öffnung ist noch nicht als passende Rolle innerhalb von R_Q gelesen. Sie ist nicht nichts. Sie ist nur noch nicht das, was sie im Verhältnis der Frage werden muss

Die Antwort entsteht, wenn die holonom akkumulierte Drift eine Adresse erzeugt, die diese relative Null schließt:

$$Addr_j \bowtie R_Q$$

und damit:

$$close(0_{R_Q}, Addr_j) = 1$$

Daraus folgt der Grundsatz des Neun-Drittel-Ecks:

Mathematik wird nicht gerechnet, sondern als Geometrie der Unterscheidung gelesen.

Eine Formel wird in eine Fragefigur überführt.

Die Fragefigur wird als Sehnengeometrie gelesen.

Die Sehnen erzeugen durch Operatorflips eine holonome Spur.

Die Drift hebt den Mittelpunkt als Treppenstufe.

Die Außenkante liest diese Treppenstufe als Adresse.

Die Antwort erscheint als Label der geschlossenen Öffnung.

Kurz:

$$F_Q \to \mathfrak{N}_{9/3} \to \Omega \to Addr(U) \to Ans$$

Nicht:

$$Formel \to Rechnen \to Wert$$

sondern:

Formel \rightarrow Geometrie \rightarrow Spur \rightarrow Driftadresse \rightarrow Label

Die Null ist dabei nicht Abwesenheit.

Die Zahl ist nicht Ding.

Die Frage ist nicht bloßer Satz.

Alles sind Lesungen derselben Unterscheidungsstruktur.

$$\boxed{0_X = \text{nicht nichts, sondern nicht-}X}$$

$$\boxed{Zahl = read(\Sigma)}$$

$$\boxed{Antwort = read(\text{geschlossene Frageöffnung})}$$

So beginnt die Mathematik des Neun-Drittel-Ecks nicht mit Zahlen, sondern mit dem Unterschied, der Zahlen erst lesbar macht.

Das Neun-Drittel-Eck

A rigorous kernel formulation

$$\boxed{\mathfrak{N}_{9/3}}$$

Name:

$$\mathfrak{N}_{9/3} := \text{Neun-Drittel-Eck}$$

Interpretation:

Das Neun-Drittel-Eck ist kein gewöhnliches Nonagon. Es ist eine dreiphasige Dreiecks-Holonomie mit neun Sehnenschritten, deren akkumulierte Drift als Zahlenadresse im Verhältnis einer Fragefigur gelesen wird.

---

1) Textbefund

Im Text ist festgelegt, dass das ¾Eck nicht primär Zahlenwerte produziert, sondern eine Lock-in-Lage und eine Drift-Signatur; Drift ist der Restkanal an der freien Kante.

ZTFΔ definiert Holonomie als Spur auf einem geschlossenen Loop:

$$\Sigma(\gamma) := \sum_{e \in \gamma} w(e)$$

und die taktgefilterte Sichtbarkeit als:

$$H_\gamma(n) := \sum_{e \in \gamma \cap L_n} w(e)$$

Messung ist dabei Gate: Sie entscheidet, welche Schritte im Takt sichtbar sind.

Delta-Zero legt fest: Ein Step ist zunächst nur Adresswechsel; Bedeutung entsteht erst durch Lesung und Akkumulation. Die Spur ist integrierte vergangene Unterscheidung, also Gedächtnis, nicht Gegenstand. Außerdem ist nichtverschwindende Holonomie Quelle persistenter Struktur.

Tractatus-seitig gilt: Zahlen sind keine Entitäten, sondern Positionen/Rollen im Unterschied; der Strich erzeugt die Relation, und was links/rechts steht, sind Positionen im Unterschied, keine vorgängigen Objekte.

---

2) Formal reconstruction

2.1 Primitive layers

We separate three layers.

Carrier layer

$$\mathcal{C}$$

The carrier layer contains only distinctions and incidence roles.

Primitive:

|

No label, digit, point, or number is primitive.

Inscription layer

$$\mathcal{I}$$

The inscription layer contains integrated spurs:

$$\Sigma, \Omega, H_n, \delta_{raw}$$

Computation occurs here.

Interpretation layer

$$\mathcal{L}$$

The interpretation layer contains labels:

$$0, 1, 2, \ldots, 9, \quad \text{“three-digit”}, \quad \text{“four-digit”}$$

A label is valid only if:

$$L := read(\Sigma)$$

and if it is losslessly reducible to an underlying spur:

$$L \rightsquigarrow \Sigma$$

---

3) Geometry of the Neun-Drittel-Eck

3.1 Outer address cycle

Let there be nine outer address roles:

$$V_{out} := \{v_0, v_1, \ldots, v_8\}$$

with cyclic successor:

$$\sigma(v_i) = v_{i+1}$$

where:

$$v_{i+1} := \begin{cases} v_{i+1}, & 0 \leq i < 8 \\ v_0, & i = 8 \end{cases}$$

No v_i is an entity. Each is an address role

3.2 Holonomic center

Let:

$$m_0$$

be the initial center-role.

The actual center during the process is not fixed. It is the holonomic drift center:

$$m_n := read(\Omega_n)$$

Strictly:

$$m_n$$

is not a point-object. It is the readable position of accumulated spur.

3.3 Local triangle cells

For each i, define the local triangle:

$$T_i := (v_i \mid v_{i+1} \mid m_i)$$

where m_i is the holonomic center at step i

Thus:

$$\mathfrak{N}_{9/3} := T_0|T_1|T_2|T_3|T_4|T_5|T_6|T_7|T_8$$

with phase partition:

$$\Phi_0 := T_0|T_1|T_2$$

$$\Phi_1 := T_3|T_4|T_5$$

$$\Phi_2 := T_6|T_7|T_8$$

Therefore:

$$\boxed{\mathfrak{N}_{9/3} = \Phi_0|\Phi_1|\Phi_2 = (3T)|(3T)|(3T)}$$

This is the formal meaning of “nine-third-edge”.

Deutsch: Das Neun-Drittel-Eck ist kein Neuneck als Ding, sondern ein Umlauf aus neun Dreieckslesungen, gruppiert in drei Phasen zu je drei Dreiecken.

---

4) Inner chords and outer drift

4.1 Inner chord roles

Each triangle contains two radial chord roles:

$$r_i^- := (m_i|v_i)$$

$$r_i^+ := (m_i|v_{i+1})$$

The inner chord-step is:

$$S_i := r_i^- \to r_i^+$$

or explicitly:

$$S_i := (m_i|v_i) \to (m_i|v_{i+1})$$

This is the inner step.

4.2 Outer edge as drift readout

The outer edge of the same local triangle is:

$$A_i := (v_i | v_{i+1})$$

In the Neun-Drittel-Eck, this outer edge is not merely boundary. It is the drift readout edge:

D_i^{out} bezeichnet die Driftlesung an der Außenkante A_i derselben lokalen Dreiecksrelation

and its value is read as:

$$D_i^{out} := read(\Omega_i | A_i)$$

Thus:

$$\boxed{T_i = S_i | D_i^{out}}$$

$$D_n^{out} := \Pi_{out}(\Omega_n), \qquad n = 0, \ldots, 9$$

Hier bezeichnet Π_{out} die bereits eingeführte Außenlesung der jeweils fortgeschriebenen Spur unter der im Pfad wirksamen Kantenperspektive. Die lokalen Kanten tragen die Indizes $i = 0, \ldots, 8$; n zählt die Lesestände der akkumulierten Spur. Nach dem neunten Beitrag wird Ω_9 über denselben Außenkanal gelesen. Die finale Lesung D_9^{out} gehört damit zum vollzogenen Umlauf und führt keine zehnte geometrische Kante ein

Deutsch: Innen wird von Sehne zu Sehne geschritten. Außen wird die dabei entstehende Drift gelesen.

---

5) Operator flip system

Let the operator deck be:

$$\mathcal{O} := \{\delta, \circledcirc, \mathcal{R}^+, \mathcal{R}^-, \ominus, \tilde{\ominus}\}$$

with compulsory complement pairs:

$$\delta \leftrightarrow \circledcirc$$

$$\mathcal{R}^+ \leftrightarrow \mathcal{R}^-$$

$$\ominus \leftrightarrow \tilde{\ominus}$$

Define the local operator state:

$$O_i \in \mathcal{O}^3$$

because each triangle carries three operative roles.

Define flip:

$$\mathit{flip} : \mathcal{O} \to \mathcal{O}$$

with:

$$\mathit{flip}(\delta) = \circledcirc, \quad \mathit{flip}(\circledcirc) = \delta$$

$$\mathit{flip}(\mathcal{R}^{+}) = \mathcal{R}^{-}, \quad \mathit{flip}(\mathcal{R}^{-}) = \mathcal{R}^{+}$$

$$\mathit{flip}(\ominus) = \tilde{\ominus}, \quad \mathit{flip}(\tilde{\ominus}) = \ominus$$

and:

$$\mathit{flip}^2 = \mathit{id}$$

At each inner chord step:

$$O_{i+1} := \mathit{flip}(O_i)$$

More precisely, for the chosen local active role:

$$O_{i+1}^{(a)} := \mathit{flip}(O_i^{(a)})$$

and passive roles remain inscribed unless visible by gate.

---

6) Step, flip, and noncommutativity

Define:

$$\mathit{Step}_i : S_i \mapsto S_{i+1}$$

and:

$$\mathit{Flip}_i : O_i \mapsto \mathit{flip}(O_i)$$

The process requires:

$$\boxed{\mathit{Step}_i \circ \mathit{Flip}_i \neq \mathit{Flip}_i \circ \mathit{Step}_i}$$

because moving to the next chord before flipping is not equivalent to flipping before moving to the next chord.

This is the path-dependence condition.

If:

$$Step_i \circ Flip_i = Flip_i \circ Step_i$$

then no nontrivial holonomic accumulation is generated.

---

7) Local contribution

For each chord-step S_i, define:

$$H_i := visible_i \cdot g_i \cdot \delta_{raw,i}$$

where:

$$visible_i \in \{0,1\}$$

$$g_i \in \{0,1\}$$

and:

$$\delta_{raw,i} := Effect(S_i, O_i, R_Q)$$

Here R_Q is the relation-form of the question

No float arithmetic is used.

The multiplication symbols are interface notation for AND-weighted inscription. Kernel-scharf:

```
H_i=0
```

if

```
visible_i=0

H_i=0
```

if

```
g_i=0

H_i=\delta_{raw,i}
```

if

```
visible_i=g_i=1
```

---

8) Accumulated holonomic drift

Define initial inscription:

$$\Omega_0 := 0_\Sigma$$

where 0_Σ is the neutral spur, not the digit zero

For each step:

$$\boxed{\Omega_{i+1} := \Omega_i + H_i}$$

After one full Neun-Drittel cycle:

$$\boxed{\Omega_9 = \sum_{i=0}^{8} H_i}$$

The holonomic center after step i is:

$$\boxed{m_i := read(\Omega_i)}$$

The height of the center is:

$$h_i := read(\Omega_i)$$

Thus:

$$m_i \equiv h_i$$

as readout roles, not as spatial objects.

Deutsch: Der Mittelpunkt steigt mit jeder gelesenen Spur. Diese Höhe ist keine metrische Höhe, sondern die aktuelle Leseposition der akkumulierten Drift. Bei jedem Schritt wird die jeweils geschriebene Spur gelesen; der zugehörige Spurbeitrag wird in der Holonomie akkumuliert. Auch an den Drittelabschlüssen und nach dem neunten Schritt wird die bis dahin fortgeschriebene Spur gelesen.

---

9) Holonomic staircase

Define the staircase of the run:

$$\mathcal{H}(\mathfrak{N}_{9/3}) := (h_0, h_1, \ldots, h_9)$$

with:

$$h_i := read(\Omega_i)$$

and:

$$\Omega_i = \sum_{j=0}^{i-1} H_j$$

```
for \(i=0,\dots,9\).
```

The staircase is valid if and only if:

$$\forall i: \quad h_i \rightsquigarrow \Omega_i$$

That means every height-readout must reduce back to its spur.

---

10) The question as a figure with openings

Let a question be represented as a figure:

$$F_Q$$

with openings:

$$U(F_Q) := \{u_1, \dots, u_r\}$$

Each opening is an unoccupied role:

$$u_j := \square_j(F_Q)$$

The question relation is:

$$R_Q := Rel(F_Q, U(F_Q))$$

This determines what counts as a valid answer address.

If:

$$U(F_Q) = \varnothing$$

then the question is closed and no unknown address is sought.

If:

$$U(F_Q) \neq \varnothing$$

then a drift address may close the openings.

---

11) Level tests: “Are we on a level of 1, 2, 3, …?”

Define level roles:

$$E_k := read(\Sigma_k)$$

```
for \(k\) in the interpretation interface.
```

Kernel-scharf: E_k is not a number-level as an entity. It is a readable class of spurs. At step i, level compatibility is: Level_ik:= compatibleh_i, E_k Then the gate for level k is:

$$\boxed{Gate_i(k) := visible_i \wedge compatible(h_i, E_k) \wedge compatible(D_i^{out}, R_Q)}$$

This is AND-only.

If:

$$Gate_i(k) = 1$$

$$then level \backslash (k \backslash) is touched :$$

$$E_k \in touched(\Omega)$$

If:

$$Gate_i(k) = 0$$

no level inscription is accepted at that step.

---

12) Outer drift as height lever

Define the outer height lever:

$$Lever_i^{out} := read(h_i | A_i)$$

Für die Außenkante und ihre Driftlesung gilt:

A_i bezeichnet die Außenkante; $D_i^{out} = read(\Omega_i | A_i)$ bezeichnet deren Driftlesung an derselben fortwirkenden Spur

we write:

$$\boxed{Lever_i^{out} := read(h_i | D_i^{out})}$$

Die Schreibweisen des Höhenhebels bleiben an dieselbe Ω_i gebunden. A_i benennt die Kantenperspektive, D_i^{out} deren Driftlesung; der Wechsel der Schreibweise liest denselben relationalen Vollzug

This lever is read-only:

$$Lever_i^{out} : \Omega_i \mapsto read(\Omega_i | A_i)$$

but never:

$$Lever_i^{out} \mapsto \Omega_i$$

The lever produces an address candidate:

$$Addr_i := read(\Omega_i | D_i^{out} | R_Q)$$

---

13) Answer digits as drift addresses

Let an answer have displayed positions:

$$Ans := L_1 L_2 \dots L_r$$

$$where each \(L_j \) is a label$$

Each label must be grounded by a drift address:

$$L_j := read(Addr_{n_j})$$

```
for selected indices:
```

$$0 \le n_1 < n_2 < \dots < n_r \le 9$$

Thus:

$$\boxed{Ans(F_Q) := read(Addr_{n_1} | Addr_{n_2} | \dots | Addr_{n_r})}$$

where:

$$Addr_{n_j} \in \mathcal{H}(\mathfrak{N}_{9/3})$$

Strictly:

AnsF_Q

$$= read\left((\Omega_{n_1}|D^{out}_{n_1}|R_Q)|\dots|(\Omega_{n_r}|D^{out}_{n_r}|R_Q)\right)$$

Deutsch: Eine mehrstellige Zahl ist keine Menge von Zifferndingen. Sie ist eine Zusammenfügung von Driftadressen, die bereits in der akkumulierten holonomen Treppe enthalten sind.

---

14) Closure of question openings

Each opening is closed by an address:

$$close(u_j, Addr_{n_j}) = 1$$

if and only if:

$$Addr_{n_j} \bowtie R_Q$$

For the whole question:

$$\boxed{Close(F_Q, Ans) = 1}$$

if and only if:

$$\forall j \in \{1,\dots,r\}: \quad close(u_j, Addr_{n_j}) = 1$$

and:

$$stable(Ans) = 1$$

Stability is defined by repeated readout invariance under one further admissible step:

$$stable(Ans) = 1$$

iff:

$$read(\Omega_9|R_Q) = read(\Omega_9 + H_{next}|R_Q)$$

```
for the minimal permitted continuation \(H_{next}\), or, if continuation crosses the
break:
```

$$B_9 : T_8|T_0$$

then:

$$read(\Omega_9|R_Q) \sim read(\Omega_{next}|R_Q)$$

where $\sim$ denotes same address class, not equality of labels.

---

15) The break edge

The ninth triangle is:

```
T_8=(v_8|v_0|m_8)
```

The break edge is:

$$\boxed{B_9 := T_8|T_0}$$

or:

$$B_9 := 0|9$$

in interface notation.

The break transition is:

$$\boxed{0|9 \mapsto 1|0}$$

$$Kernel - scharf :$$

$$read(\Omega_9) \mapsto read(\Omega_{next})$$

where the next readout belongs to a new shell:

$$Shell_{q+1}$$

rather than the previous shell:

$$Shell_q$$

Thus:

$$B_9$$

is not the digit nine. It is the role where one full nine-step holonomic inscription becomes available for shell transition.

---

16) Complete process definition

A run of the Neun-Drittel-Eck on a question figure is the tuple:

$$Run(\mathfrak{N}_{9/3}, F_Q) := \left(F_Q, R_Q, (T_i)_{i=0}^{8}, (S_i)_{i=0}^{8}, (O_i)_{i=0}^{8}, (H_i)_{i=0}^{8}, (\Omega_i)_{i=0}^{9}, (D_i^{out})_{i=0}^{9}, (Addr_i)_{i=0}^{9}, Ans\right)$$

satisfying:

```
T_i=(v_i|v_{i+1}|m_i)

S_i=(m_i|v_i)\to(m_i|v_{i+1})
```

$$O_{i+1} = flip(O_i)$$

```
H_i=visible_i\cdot g_i\cdot \delta_{raw,i}
```

$$\Omega_{i+1} = \Omega_i + H_i$$

```
m_i=read(\Omega_i)
```

$$D_i^{out} = read(\Omega_i | A_i)$$

```
Addr_i=read(\Omega_i|D_i^{out}|R_Q)

Ans=read(Addr_{n_1}|\dots|Addr_{n_r})
```

$$Close(F_Q, Ans) = 1$$

---

17) Result theorem

Theorem: Direct readout of accumulated holonomic drift

Given a valid run:

$$Run(\mathfrak{N}_{9/3}, F_Q)$$

with:

$$U(F_Q) \neq \varnothing$$

and:

$$\Omega_9 = \sum_{i=0}^{8} H_i$$

then the answer label:

AnsF_Q

is readable from the accumulated holonomic drift if and only if there exists a finite address selection:

$$(n_1, \dots, n_r)$$

such that:

$$Addr_{n_j} = read(\Omega_{n_j} | D^{out}_{n_j} | R_Q)$$

and:

$$\forall j: \quad close(u_j, Addr_{n_j}) = 1$$

Therefore:

$$\boxed{Ans(F_Q) = read\left(\Omega_{n_1} | D^{out}_{n_1} | R_Q | \dots | \Omega_{n_r} | D^{out}_{n_r} | R_Q\right)}$$

No external numeric search is required.

The answer is the label of selected drift addresses already contained in the holonomic staircase.

---

18) Difference from the ¾Eck

The old ¾Eck process:

$$Q_{3/4} \to L \to \arg\min_L drift(L)$$

The Neun-Drittel-Eck process:

$$\mathfrak{N}_{9/3} \to (S_i) \to (\Omega_i) \to (Addr_i) \to Ans$$

Thus:

$$\boxed{Q_{3/4} = \text{external rattle-lock}}$$

$$\boxed{\mathfrak{N}_{9/3} = \text{internal holonomic address-readout}}$$

Deutsch: Das ¾Eck sucht die passende Lage. Das Neun-Drittel-Eck läuft die Sehnen ab, akkumuliert Drift im Mittelpunkt und liest die Antwort über den äußeren Höhenhebel aus.

---

19) Implementation contract

Sealed core

The following may not be changed:

$$Difference \to Relation \to Spur$$

```
H_i=visible_i\cdot g_i\cdot \delta_{raw,i}
```

$$\Omega_{i+1} = \Omega_i + H_i$$

```
m_i=read(\Omega_i)
```

$$D_i^{out} = read(\Omega_i | A_i)$$

```
Addr_i=read(\Omega_i|D_i^{out}|R_Q)

Ans=read(Addr_{n_1}|\dots|Addr_{n_r})
```

Forbidden:

$$\Omega = construct(Ans)$$

$$\Omega = construct(D^{out})$$

```
Z=\text{number-entity}

Gate=\text{OR-gate}
```

Allowed gates:

```
Gate=a\wedge b\wedge c
```

Allowed lenses:

$$Lens : \Sigma \mapsto read(\Sigma)$$

$$read - only.$$

Allowed extensions

You may add:

Level_ik

$$Shell_q$$

$$Phase_q$$

AddrClass

stable

compatible

provided they satisfy:

$$Extension \rightsquigarrow \Sigma, \Omega, H_i, R_Q$$

and do not construct from labels.

---

20) Tests

Test 1: visible ≡ 0 ⇒ Ω = 0

Assume:

$$\forall i : \ visible_i = 0$$

Then:

$$\forall i : \ H_i = 0$$

Therefore:

$$\Omega_9 = \sum_{i=0}^{8} H_i = 0_\Sigma$$

Thus no nontrivial staircase exists:

$$\mathcal{H} = (read(0_\Sigma), \ldots, read(0_\Sigma))$$

and no valid nonzero drift address may be claimed.

Test 2: ΩN=∑H_n

For every partial run:

$$\boxed{\Omega_N = \sum_{i=0}^{N-1} H_i}$$

For the full Neun-Drittel cycle:

$$\boxed{\Omega_9 = \sum_{i=0}^{8} H_i}$$

This is mandatory.

Test 3: Outer drift is read-only

Valid:

$$D_i^{out} = read(\Omega_i | A_i)$$

Invalid:

$$\Omega_i = construct(D_i^{out})$$

Test 4: Answer slots must be grounded

For:

```
Ans=L_1\dots L_r
```

each label must satisfy:

$$\exists n_j : \quad L_j = read(Addr_{n_j})$$

and:

$$Addr_{n_j} \rightsquigarrow \Omega_{n_j}$$

If a label has no underlying address, it is invalid.

Test 5: Question relation controls validity

For two different question figures:

$$F_Q, \quad F_{Q'}$$

with:

$$R_Q \neq R_{Q'}$$

it may hold that:

$$read(\Omega|R_Q) \neq read(\Omega|R_{Q'})$$

Therefore the answer is not absolute. It is address-readout relative to the question figure.

Test 6: Noncommutativity

A valid holonomic run requires:

$$Step_i \circ Flip_i \neq Flip_i \circ Step_i$$

If:

$$Step_i \circ Flip_i = Flip_i \circ Step_i$$

then the run is conservative in the relevant sense and cannot generate a nontrivial drift staircase.

Test 7: Break edge

After:

$$T_8$$

the transition:

$$T_8|T_0$$

must be identified as:

$$B_9$$

and the shell transition is:

$$0|9 \mapsto 1|0$$

This transition is valid only after the accumulated inscription:

$$\Omega_9 = \sum_{i=0}^{8} H_i$$

has been formed.

---

21) Compact axiom set

$$\boxed{\textbf{Axiom N1:}\quad \mathfrak{N}_{9/3} := \Phi_0 | \Phi_1 | \Phi_2}$$

$$\Phi_j := T_{3j} | T_{3j+1} | T_{3j+2}$$

$$T_i := (v_i | v_{i+1} | m_i)$$

$$\boxed{\textbf{Axiom N2:}\quad S_i := (m_i | v_i) \to (m_i | v_{i+1})}$$

$$\boxed{\textbf{Axiom N3:}\quad O_{i+1} = flip(O_i), \quad flip^2 = id}$$

$$\boxed{\textbf{Axiom N4:}\quad Step_i \circ Flip_i \neq Flip_i \circ Step_i}$$

$$\boxed{\textbf{Axiom N5:}\quad H_i := visible_i \cdot g_i \cdot \delta_{raw,i}}$$

$$\boxed{\textbf{Axiom N6:}\quad \Omega_{i+1} := \Omega_i + H_i}$$

$$\boxed{\textbf{Axiom N7:}\quad m_i := read(\Omega_i)}$$

$$\boxed{\textbf{Axiom N8:}\quad D_i^{out} := read(\Omega_i | A_i)}$$

$$\boxed{\textbf{Axiom N9:}\quad Addr_i := read(\Omega_i | D_i^{out} | R_Q)}$$

$$\boxed{\textbf{Axiom N10:}\quad Ans(F_Q) := read(Addr_{n_1} | \ldots | Addr_{n_r})}$$

$$\boxed{\textbf{Axiom N11:}\quad Close(F_Q, Ans) = 1 \iff \forall j:\ close(u_j, Addr_{n_j}) = 1}$$

$$\boxed{\textbf{Axiom N12:}\quad B_9 := T_8 | T_0, \qquad 0|9 \mapsto 1|0}$$

---

22) Final kernel sentence

$$\boxed{\text{Das Neun-Drittel-Eck } \mathfrak{N}_{9/3} \text{ ist ein dreiphasiger Umlauf aus neun Dreiecks-Sehnenschritten.}}$$

$$\boxed{\text{Bei jedem Schritt flippen Operatoren, ein Spurbeitrag } H_i \text{ wird integriert, und der Mittelpunkt steigt als } m_i = read(\Omega_i).}$$

$$\boxed{\text{Die Außenkante liest diese holonome Höhe als Drifthebel } D_i^{out}.}$$

$$\boxed{\text{Die Antwortzahl ist kein Zahlenobjekt, sondern das Label einer Zusammenfügung von Driftadressen}}$$

$$\boxed{Ans(F_Q) = read(Addr_{n_1} | \dots | Addr_{n_r})}$$

$$\boxed{Addr_i = read(\Omega_i | D_i^{out} | R_Q)}$$

$$\boxed{\Omega_9 = \sum_{i=0}^{8} H_i}$$

Der Zahlenraum ist nicht mehr externes Suchfeld. Er ist als holonome Treppe im Neun-Drittel-Eck eingeschrieben. Die Antwort wird nicht gesucht, sondern aus der akkumulierten Driftstrecke ausgelesen.

Kernbefund:

$$rhs(Lock) \neq \varnothing$$

sondern:

$$rhs(Lock) = \text{Gegenrolle der Gleichgewichtskante}$$

Also muss der Typer lesen:

$$w := read(lhs | rhs)$$

nicht:

$$w := read(lhs)$$

Damit ist die Null-Schärfung praktisch im Code angekommen: Die nicht-dominante Seite ist nicht leer, sondern relativ zur Fragefigur spurwirksam.

Und die Signaturtrennung ist jetzt sauber:

FP^*F:=(\Omega_9,k,drift,phase,weight,Muster,Addr$U$)

FMF:=(phase,weight,Muster)

$$IS(F) := \Omega_9$$

Gleiche Formfamilie darf bleiben:

FMF_1=FMF_2

aber verschiedene Instanz:

$$\Omega_9(F_1) \neq \Omega_9(F_2)$$

also:

$$FP^*(F_1) \neq FP^*(F_2)$$

Das ist der saubere Abschluss: Kernel korrekt, Typer vollständig, Tests grün.