ACF 79 — Verification Dossier

The Source

Substitution-Initiated Self-Separation of the Substituted-Chrysene Stock from a Polyaromatic Bulk, and the Supply It Delivers to the Protocell

Charles D. Schaper, Ph.D., Schaper Systems Press

Annals of the Chrysene Formalism, Vol. 1, Issue 4

Verification script: `verify_A79.py` Lean proof: `acf-79-proof.lean`

Certification Summary

Item	Value
Script	`verify_A79.py`
Total checks	88
Passed	**88**
Failed	**0**
Exit code	`0`
Dependencies	Python standard library only
Lean proof file	`acf-79-proof.lean`
Lean version	`4.12.0` (core only, no Mathlib)
Lean theorems	55
Theorems with no axioms	54
Theorems using `propext`	1
`sorry`	0
`native_decide`	0

Purpose and Scope

This dossier accompanies Article 79, "The Source," which derives a source of substituted-chrysene stock that separates itself from a polyaromatic bulk as it is made, and matches the supply it delivers to the demand of the protocell.

The verification script `verify_A79.py` executes 88 independent checks in fifteen blocks (T1–T15), covering the imported demand, the arm ladder and the Ordering theorem, the exit distribution under the adopted kinetics, the design equation, matched efficiency and its lower bound, the cooperativity trade-off, the initiator surplus, the slow-completion contrast case, throughput, the two-zone Bateman variant, arm-word orbit counts, arm-efficiency sensitivity, carbon per admitted unit and the molecular formulae, column impurity against the Perron limit, and the scission, residence and registry certificates. All 88 checks passed with exit code 0.

Exact where the claim is finite. Blocks T1–T9 and T11–T13 are computed in exact rational arithmetic (`fractions.Fraction`); no floating point enters them, so the equalities they assert are equalities and not agreements to a tolerance. Floating point is confined to T10 (a transcendental root), the Perron limit in T14, and the registry exponentials in T15; each such block is labelled FLOATING in its header and in the transcript.

The companion Lean 4 proof file `acf-79-proof.lean` provides kernel-checked formal certificates for 55 of the finite arithmetic claims.

Block Index

Block	Topic	Checks
T1	Imported demand (R_D, R_T, ratios, rounding)	8
T2	Arm ladder and the Ordering theorem (Thm 5.1)	5
T3	Exit distribution under the adopted kinetics	6
T4	The design equation (Thm 7.3)	4
T5	Matched efficiency and its lower bound	6
T6	Cooperativity trade-off	6
T7	Initiator surplus	3
T8	Contrast case: slow completion	6
T9	Throughput and efficiency	4
T10	Two-zone variant: Bateman crossing (FLOATING)	4
T11	Arm words within Article 77's families	6
T12	Arm-efficiency sensitivity	4
T13	Carbon per admitted unit and molecular formulae	10
T14	Column impurity vs the Perron limit (FLOATING LIMIT)	5
T15	Scission, residence, registry (PART FLOATING)	11
	Total	**88**

Scope of Verification

What is verified.

- The imported demand of Article 65 Thm 8.5 with the Article 78 ICL bilayer: $R_D = 42\,354$ at $\alpha = 1/10$, $R_T = L + n_{\mathrm{ICL}} = 208$, and the ratio $R_D/R_T = 1629/8$ in lowest terms.
- Hypothesis H_{add}'s geometric ladder $P_n = P_0 q^n$ at $q = 10$, and the conclusion of Theorem 5.1 (Ordering), $P_0 < P_1 < P_2 < P_3 < P_4$.
- The exit distribution computed two independent ways — a closed form and the fundamental matrix of the absorbing chain — agreeing exactly, summing to one exactly, and satisfying the design equation $p_2/p_4 = R_D/R_T$ exactly.
- Matched efficiency $\eta = p_2 + p_4$ against its stated lower bound, and the exact identity $\eta_{\mathrm{full}} - \eta = c_I/\Phi$.
- The cooperativity identity $k_4/k_3 = q(R_D/R_T)/(x(1+x))$ at three completion ratios $x = e_3/k_4$, each re-matched to the demand.

- Burnside orbit counts for arm words in Article 77's three families, checked against brute-force enumeration at alphabet sizes $m = 2, 3, 4$.
- Carbon per admitted unit $C = 18 + 2n$; the molecular formulae of the di- and tetra-(hydroxyethoxy) units, the tetra-ol and the bis(oxalate) acid; the invariance of carbon under phosphorylation and under oxalate reduction; the cycle carbon 937 196 and the lipid coproduct 26 708.
- The exact finite-column transfer-matrix impurity sums against the Perron limit by Richardson extrapolation.
- The M-matrix certificate for the open source: every absorption time strictly positive; the admissible and waste masses partitioning the exit distribution.

What is not verified. Existential and qualitative claims not expressible as closed-form numerical statements fall outside this harness. The rate parameters of blocks T3–T9 are *stipulated nondimensional illustrations*, not measured chemistry: the harness certifies that the stated numbers follow from the stated rates, not that the rates obtain in any laboratory. The regioselectivity of the bromination sequence is an open premise of the article and is not addressed here. Continuous functional-analytic claims about the Chrysene formalism C*-algebra are outside the scope of both this harness and the Lean file.

Script Listing: `verify_A79.py`

verify_A79.py — complete source

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
verify_A79.py -- verification harness for Article 79 of the
Annals of the Chrysene Formalism:

    "The Source: Substitution-Initiated Self-Separation of the
     Substituted-Chrysene Stock from a Polyaromatic Bulk, and the
     Supply It Delivers to the Protocell"

Standard library only.  Exact rational arithmetic (fractions.Fraction)
wherever the claim in the paper is finite; floating point only where an
exponential, a square root or a limit enters, and every such block is
labelled FLOATING in its header.

Each check is a named assertion with an identifier Tn-mm.  The script
prints PASS or FAIL per check, a final tally, and exits non-zero if any
check fails, so it is usable as a release gate.

    python3 verify_A79.py        summary
    python3 verify_A79.py -v     with the computed values

Blocks
  T1   Imported demand (Art. 65 Thm 8.5; Art. 78; Art. 77 Prop 3.12)
  T2   The arm ladder and the Ordering theorem (Thm 5.1)
  T3   Exit distribution under the adopted kinetics
  T4   The design equation (Thm 7.3)
  T5   Matched efficiency and its lower bound
  T6   Cooperativity trade-off
  T7   Initiator surplus
  T8   Contrast case: slow completion
  T9   Throughput and efficiency
  T10  Two-zone variant: Bateman crossing            FLOATING
  T11  Arm words within Article 77's families
  T12  Arm-efficiency sensitivity
  T13  Carbon per admitted unit and molecular formulae
  T14  Column impurity: exact sums against the Perron limit  FLOATING
  T15  Scission certificates, residence, registry     part FLOATING
"""

import itertools
import math
import sys
from fractions import Fraction as F

VERBOSE = "-v" in sys.argv or "--verbose" in sys.argv

PASSED = 0
FAILED = 0
_CURRENT = [""]


def block(tag, title, note=""):
```

```
    print()
    print("=" * 72)
    print("  %s  %s%s" % (tag, title, ("   [%s]" % note) if note else "")
        )
    print("=" * 72)
    _CURRENT[0] = tag


def check(ident, label, condition, detail=""):
    global PASSED, FAILED
    if condition:
        PASSED += 1
        print("[PASS] %-9s %s" % (ident, label))
    else:
        FAILED += 1
        print("[FAIL] %-9s %s" % (ident, label))
    if detail and (VERBOSE or not condition):
        print("    %s" % detail)


# ======================================================================
#  Corpus constants
# ======================================================================
L = 200                    # genome length, bases
N = F(252000)              # Article 65 pool constant
nu_lip = 66                # lipids per base
nICL = 8                   # pore T0 units, once per founder
ellICL = 77                # bilayer lipids of the minimal ICL (Art. 78
    Prop 9.1)
cI = 1                     # initiators drawn per strand-period
q = F(10)                  # arm multiplier of H_add

# export ladder e_n = e_0 q^n, with e_2 = 1 fixing the time unit
e = [F(1, 100) * q ** n for n in range(5)]


def R_D_base(alpha):
    """Article 65 Thm 8.5."""
    return 5 * L + alpha / (1 - alpha) * N + nu_lip * L


def R_D(alpha):
    """Thm 8.5 plus the Article 78 ICL bilayer: two D units per route-A
        lipid."""
    return R_D_base(alpha) + 2 * ellICL


RD10 = R_D(F(1, 10))
RD02 = R_D(F(2, 100))
RT = F(L + nICL)


# ======================================================================
#  Markov-chain helpers (exact)
# ======================================================================
def exit_closed_form(k, ee):
    kk = list(k) + [F(0)]
```

```
        p, reach = [], F(1)
        for n in range(5):
            tot = ee[n] + kk[n]
            p.append(reach * ee[n] / tot)
            reach *= kk[n] / tot
        return p


def solve(A, b):
    """Exact Gaussian elimination over Fractions."""
    n = len(A)
    M = [list(row) + [b[i]] for i, row in enumerate(A)]
    for c in range(n):
        piv = next(r for r in range(c, n) if M[r][c] != 0)
        M[c], M[piv] = M[piv], M[c]
        for r in range(n):
            if r != c and M[r][c] != 0:
                f = M[r][c] / M[c][c]
                M[r] = [x - f * y for x, y in zip(M[r], M[c])]
    return [M[i][n] / M[i][i] for i in range(n)]


def exit_fundamental(k, ee):
    """Same distribution via the fundamental matrix -- an independent
       route."""
    kk = list(k) + [F(0)]
    Qt = [[F(0)] * 5 for _ in range(5)]
    R = [[F(0)] * 5 for _ in range(5)]
    for n in range(5):
        tot = ee[n] + kk[n]
        if n < 4:
            Qt[n][n + 1] = kk[n] / tot
        R[n][n] = ee[n] / tot
    AT = [[(F(1) if i == j else F(0)) - Qt[j][i] for j in range(5)] for i
         in range(5)]
    N0 = solve(AT, [F(1), F(0), F(0), F(0), F(0)])
    return [sum(N0[m] * R[m][n] for m in range(5)) for n in range(5)]


def mean_residence(k, ee):
    kk = list(k) + [F(0)]
    t, reach = F(0), F(1)
    for n in range(5):
        tot = ee[n] + kk[n]
        t += reach / tot
        reach *= kk[n] / tot
    return t


print("=" * 72)
print("  verify_A79.py -- Article 79, 'The Source'")
print("  Annals of the Chrysene Formalism, Vol. 1, Issue 5")
print("=" * 72)

# ======================================================================
block("T1", "Imported demand")
# ======================================================================
```

```
    check("T1-01", "R_D base at alpha=1/10 is 42 200 units (Art. 65 Thm 8.5)"
        ,
          R_D_base(F(1, 10)) == 42200, "R_D_base = %s" % R_D_base(F(1, 10)))
    check("T1-02", "R_D at alpha=1/10 adds 2*ellICL: 42 354",
          RD10 == 42354, "R_D = %s = %s + 2*%d" % (RD10, R_D_base(F(1, 10)),
              ellICL))
    check("T1-03", "R_D at alpha=2/100 is 136478/7",
          RD02 == F(136478, 7), "R_D(0.02) = %s" % RD02)
    check("T1-04", "R_T founder draw is L + nICL = 208",
          RT == L + nICL, "R_T = %s" % RT)
    check("T1-05", "demand ratio R_D/R_T reduces to 1629/8",
          RD10 / RT == F(1629, 8) and math.gcd(42354, 208) == 26,
          "R_D/R_T = %s, gcd = %d" % (RD10 / RT, math.gcd(42354, 208)))
    check("T1-06", "R_D/R_T rounds to 204 (2*|R_D - 204 R_T| < R_T)",
          2 * abs(42354 - 204 * 208) < 208,
          "2*%d = %d < 208" % (abs(42354 - 204 * 208), 2 * abs(42354 - 204 *
              208)))
    check("T1-07", "R_D(0.02)/R_T rounds to 94",
          2 * abs(136478 - 94 * 7 * 208) < 7 * 208,
          "2*%d < %d" % (abs(136478 - 94 * 1456), 1456))
    check("T1-08", "the D demand exceeds the founder pore draw 100-fold",
          100 * RT < RD10, "100 R_T = %s < R_D = %s" % (100 * RT, RD10))

    # ====================================================================
    block("T2", "The arm ladder and the Ordering theorem (Thm 5.1)")
    # ====================================================================
    check("T2-01", "H_add gives a geometric ladder e_n = e_0 q^n, q = 10",
          all(e[n + 1] == e[n] * q for n in range(4)),
          "e = %s" % [str(x) for x in e])
    check("T2-02", "q exceeds unity, so every arm raises the export rate",
          q > 1, "q = %s" % q)
    check("T2-03", "Ordering (Thm 5.1): P_0 < P_1 < P_2 < P_3 < P_4",
          all(e[n] < e[n + 1] for n in range(4)),
          "strictly increasing across all four steps")
    check("T2-04", "the ladder spans q^4 = 10 000 from P_0 to P_4",
          e[4] / e[0] == q ** 4, "P_4/P_0 = %s" % (e[4] / e[0]))
    check("T2-05", "e_2 = 1 fixes the time unit",
          e[2] == 1, "e_2 = %s" % e[2])

    # ====================================================================
    block("T3", "Exit distribution under the adopted kinetics")
    # ====================================================================
    k1 = k2 = F(100)            # 6 and 12 near-simultaneous
    k4 = F(100)                 # short delay 3,6,12 -> 3,6,9,12
    k3 = e[2] * (RT / RD10) * (1 + e[3] / k4)      # design equation
    kA = [k1, k2, k3, k4]
    pA = exit_closed_form(kA, e)
    pF = exit_fundamental(kA, e)

    check("T3-01", "closed form agrees with the fundamental matrix exactly",
          pA == pF, "max difference = %s" % max(abs(a - b) for a, b in zip(pA
              , pF)))
    check("T3-02", "the exit distribution sums to one exactly",
          sum(pA) == 1, "sum = %s" % sum(pA))
    check("T3-03", "every exit probability is strictly positive",
          all(p > 0 for p in pA), "min = %s" % min(pA))
    check("T3-04", "the disubstituted channel carries the mass: p_2 > 0.99",
```

```
        pA[2] > F(99, 100), "p_2 = %.6f" % float(pA[2]))
check("T3-05", "p_3 / p_4 = 1/q: the trisubstituted channel is a tenth",
        pA[3] / pA[4] == 1 / q, "p_3/p_4 = %s" % (pA[3] / pA[4]))
check("T3-06", "mean residence is finite and near one time unit",
        F(1) < mean_residence(kA, e) < F(11, 10),
        "mean residence = %.3f" % float(mean_residence(kA, e)))

# ======================================================================
block("T4", "The design equation (Thm 7.3)")
# ======================================================================
check("T4-01", "p_2 / p_4 equals the demand ratio R_D / R_T exactly",
        pA[2] / pA[4] == RD10 / RT,
        "p_2/p_4 = %s = R_D/R_T" % (pA[2] / pA[4]))
check("T4-02", "cross-multiplied: p_2 R_T = p_4 R_D",
        pA[2] * RT == pA[4] * RD10, "both = %s" % (pA[2] * RT))
check("T4-03", "k_3 solves the design equation",
        k3 == e[2] * (RT / RD10) * (1 + e[3] / k4),
        "k_3 = %s = %.6g" % (k3, float(k3)))
check("T4-04", "k_3 is far below k_1, k_2, k_4: the third arm is the gate
    ",
        k3 < k1 / 1000, "k_3 = %.6g, k_1 = %s" % (float(k3), k1))

# ======================================================================
block("T5", "Matched efficiency and its lower bound")
# ======================================================================
etaA = pA[2] + pA[4]
bound = 1 - e[0] / k1 - e[1] / k2 - (RT / (RD10 + RT)) * (e[3] / k4)
PhiA = RD10 / pA[2]
etaFull = (RD10 + RT + cI) / PhiA

check("T5-01", "matched efficiency eta = p_2 + p_4",
        etaA == pA[2] + pA[4], "eta = %.6f" % float(etaA))
check("T5-02", "eta meets the stated lower bound",
        etaA >= bound, "eta = %.6f >= bound = %.6f" % (float(etaA), float(
            bound)))
check("T5-03", "eta lies strictly below one: the source is not perfect",
        etaA < 1, "1 - eta = %.3e" % float(1 - etaA))
check("T5-04", "eta exceeds 0.998",
        etaA > F(998, 1000), "eta = %.6f" % float(etaA))
check("T5-05", "counting the initiator as delivered raises efficiency by
    "
                "exactly c_I / Phi",
        etaFull > etaA and etaFull - etaA == cI / PhiA,
        "eta_full - eta = %.3e = c_I/Phi = %.3e"
        % (float(etaFull - etaA), float(cI / PhiA)))
check("T5-06", "throughput Phi = R_D / p_2 is near 42 630",
        42600 < PhiA < 42700 and PhiA == RD10 / pA[2],
        "Phi = %.2f" % float(PhiA))

# ======================================================================
block("T6", "Cooperativity trade-off")
# ======================================================================
check("T6-01", "k_4 / k_3 = 203625/11, floor 18 511",
        k4 / k3 == F(203625, 11) and 203625 // 11 == 18511,
        "k_4/k_3 = %s = %.1f" % (k4 / k3, float(k4 / k3)))

for ratio, nm, ident in [(F(1, 10), "1/10", "T6-02"),
```

```
                            (F(1), "1", "T6-03"),
                            (F(10), "10", "T6-04")]:
        k4x = e[3] / ratio
        k3x = e[2] * (RT / RD10) * (1 + ratio)
        px = exit_closed_form([k1, k2, k3x, k4x], e)
        coop = k4x / k3x
        ok = (px[2] / px[4] == RD10 / RT
              and coop == q * (RD10 / RT) / (ratio * (1 + ratio)))
        check(ident,
              "at completion ratio e_3/k_4 = %s the match holds and "
              "k_4/k_3 = q(R_D/R_T)/(x(1+x))" % nm,
              ok, "k_4/k_3 = %.4g, p_3 = %.4f%%" % (float(coop), float(px[3])
                   * 100))

    check("T6-05", "the cooperativity requirement falls as completion slows",
          (q * (RD10 / RT) / (F(1, 10) * (1 + F(1, 10)))
           > q * (RD10 / RT) / (F(10) * (1 + F(10)))),
          "x=0.1 needs %.4g, x=10 needs %.4g"
          % (float(q * (RD10 / RT) / (F(1, 10) * F(11, 10))),
             float(q * (RD10 / RT) / (F(10) * F(11)))))
    check("T6-06", "the I-loss term grows with the completion ratio",
          ((RT / (RD10 + RT)) * F(1, 10) < (RT / (RD10 + RT)) * F(10)),
          "loss(x=0.1) = %.4f%%, loss(x=10) = %.4f%%"
          % (float((RT / (RD10 + RT)) * F(1, 10)) * 100,
             float((RT / (RD10 + RT)) * F(10)) * 100))

    # ====================================================================
    block("T7", "Initiator surplus")
    # ====================================================================
    Isurplus = PhiA * pA[3]
    check("T7-01", "initiator made per period is 104/5 = 20.8",
          Isurplus == F(104, 5), "Phi p_3 = %s = %.1f" % (Isurplus, float(
              Isurplus)))
    check("T7-02", "the initiator drawn does not exceed the initiator made",
          cI <= Isurplus, "c_I = %d <= %.1f" % (cI, float(Isurplus)))
    check("T7-03", "the surplus is at least twentyfold",
          Isurplus >= 20 * cI, "ratio = %.1f" % float(Isurplus / cI))

    # ====================================================================
    block("T8", "Contrast case: slow completion")
    # ====================================================================
    kB = [F(2), F(1), F(2, 1000), F(1, 1000)]
    pB = exit_closed_form(kB, e)
    check("T8-01", "the contrast distribution also sums to one",
          sum(pB) == 1 and pB == exit_fundamental(kB, e), "sum = %s" % sum(pB
              ))
    check("T8-02", "p_2 falls to about 0.90 when completion is slow",
          F(89, 100) < pB[2] < F(91, 100), "p_2 = %.3f" % float(pB[2]))
    check("T8-03", "p_4 collapses below 1e-6",
          pB[4] < F(1, 10 ** 6), "p_4 = %.3e" % float(pB[4]))
    check("T8-04", "p_4 / p_3 = k_4 / e_3 under slow completion",
          pB[4] / pB[3] == kB[3] / e[3], "ratio = %.2e" % float(pB[4] / pB
              [3]))
    eps = F(1, 10)
    check("T8-05", "axis window threshold (1-eps)^2/(q eps^2) = 81/10 at eps
        =0.1",
          (1 - eps) ** 2 / (q * eps * eps) == F(81, 10),
```

```
          "threshold = %s = %.2f" % ((1 - eps) ** 2 / (q * eps * eps),
                                     float((1 - eps) ** 2 / (q * eps * eps)))
                                        )
    check("T8-06", "completion threshold q(1-eps)/eps = 90 at eps=0.1",
          q * (1 - eps) / eps == 90, "threshold = %s" % (q * (1 - eps) / eps)
              )

    # ======================================================================
    block("T9", "Throughput and efficiency")
    # ======================================================================
    PhiB = max(RD10 / pB[2], RT / pB[4])
    etaB = (RD10 + RT) / PhiB
    check("T9-01", "the contrast case needs a vastly larger throughput",
          PhiB > 10000 * PhiA, "Phi_B/Phi_A = %.3g" % float(PhiB / PhiA))
    check("T9-02", "contrast efficiency collapses below 1e-4",
          etaB < F(1, 10000), "eta_B = %.2e" % float(etaB))
    check("T9-03", "adopted efficiency exceeds the contrast by over four
        decades",
          etaA / etaB > 10000, "ratio = %.3g" % float(etaA / etaB))
    check("T9-04", "the adopted throughput is finite and near 42 630",
          42000 < PhiA < 43000, "Phi_A = %.2f" % float(PhiA))

    # ======================================================================
    block("T10", "Two-zone variant: Bateman crossing", "FLOATING")
    # ======================================================================


    def bateman(a, b, t):
        Dd = math.exp(-a * t)
        Ii = a / (b - a) * (math.exp(-a * t) - math.exp(-b * t))
        return Dd, Ii, 1.0 - Dd - Ii


    def crossing(a, b):
        lo, hi = 1e-12, 1.0
        while bateman(a, b, hi)[2] < bateman(a, b, hi)[1]:
            hi *= 2
        for _ in range(200):
            mid = 0.5 * (lo + hi)
            _, Ii, Tt = bateman(a, b, mid)
            lo, hi = (mid, hi) if Tt < Ii else (lo, mid)
        return hi


    lo, hi = 0.5, 3.0
    for _ in range(200):
        mid = 0.5 * (lo + hi)
        lo, hi = (mid, hi) if mid - 2 * (1 - math.exp(-mid)) < 0 else (lo,
            mid)
    xstar = hi
    check("T10-01", "x* solves x = 2(1 - e^-x), x* approx 1.5936",
          abs(xstar - 1.5936) < 5e-4, "x* = %.6f" % xstar)
    check("T10-02", "x* is the unique positive root: the residual vanishes",
          abs(xstar - 2 * (1 - math.exp(-xstar))) < 1e-9,
          "residual = %.3e" % abs(xstar - 2 * (1 - math.exp(-xstar))))
    rows = []
    for r in [10, 50, 200, 1000, 10000]:
```

```
        tc = crossing(1.0, float(r))
        Dd, _, _ = bateman(1.0, float(r), tc)
        rows.append((r, 1 - Dd))
    check("T10-03", "r*delta converges to x* as r grows",
          abs(rows[-1][0] * rows[-1][1] - xstar) < 1e-3,
          "r=10000: r*delta = %.5f vs x* = %.5f"
          % (rows[-1][0] * rows[-1][1], xstar))
    check("T10-04", "the convergence is first order: error*r is bounded",
          all(abs(r * d - xstar) * r < 5.0 for r, d in rows[1:]),
          "max error*r = %.3f" % max(abs(r * d - xstar) * r for r, d in rows
              [1:]))

    # ============================================================================
    block("T11", "Arm words within Article 77's families")
    # ============================================================================


    def orbit_counts(m):
        """D on {6,12} (swap), I on {3,6,12} (free), T on {3,6,9,12} (iota)."
           ""
        return m * (m + 1) // 2, m ** 3, (m ** 4 + m ** 2) // 2


    def brute(m):
        al = range(m)
        Dset = {min((a, b), (b, a)) for a, b in itertools.product(al, al)}
        Tset = set()
        for w in itertools.product(al, repeat=4):
            img = (w[2], w[3], w[0], w[1])          # iota: 3<->9, 6<->12
            Tset.add(min(w, img))
        return len(Dset), m ** 3, len(Tset)


    for m in (2, 3, 4):
        check("T11-0%d" % (m - 1),
              "Burnside counts agree with brute force at m=%d" % m,
              orbit_counts(m) == brute(m),
              "closed = %s, brute = %s" % (orbit_counts(m), brute(m)))
    D2, I2, T2 = orbit_counts(2)
    D3, I3, T3 = orbit_counts(3)
    check("T11-04", "two arm types give 21 words: 3 + 8 + 10",
          (D2, I2, T2) == (3, 8, 10) and D2 + I2 + T2 == 21,
          "%d + %d + %d = %d" % (D2, I2, T2, D2 + I2 + T2))
    check("T11-05", "three arm types give 78 words: 6 + 27 + 45",
          (D3, I3, T3) == (6, 27, 45) and D3 + I3 + T3 == 78,
          "%d + %d + %d = %d" % (D3, I3, T3, D3 + I3 + T3))
    check("T11-06", "the tetrasubstituted count is the Burnside average",
          all(2 * orbit_counts(m)[2] == m ** 4 + m ** 2 for m in (2, 3, 4)),
          "2T(m) = m^4 + m^2 for m = 2,3,4")

    # ============================================================================
    block("T12", "Arm-efficiency sensitivity")
    # ============================================================================
    prev = None
    for pe, nm, ident in [(F(99, 100), "0.99", "T12-01"),
                          (F(95, 100), "0.95", "T12-02"),
                          (F(9, 10), "0.90", "T12-03")]:
```

```
        ph = 1 - pe
        accD = 1 - ph ** 2
        accT = pe ** 4
        eta_arm = (RD10 + RT) / (RD10 / accD + RT / accT)
        ok = eta_arm < 1 and (prev is None or eta_arm < prev)
        check(ident, "arm efficiency at per-arm fidelity %s is below one "
                     "and falls with fidelity" % nm,
              ok, "eta_arm = %.4f" % float(eta_arm))
        prev = eta_arm
    check("T12-04", "even at fidelity 0.90 the arm efficiency stays above
        0.98",
          prev > F(98, 100), "eta_arm(0.90) = %.4f" % float(prev))

    # ====================================================================
    block("T13", "Carbon per admitted unit and molecular formulae")
    # ====================================================================


    def formula(nh, ne, npp, no=0):
        """Neutral formula with h, e, phosphorylated-e and oxalate arms."""
        return (18 + 2 * ne + 2 * npp + 2 * no,
                12 + 4 * ne + 5 * npp,
                nh + 2 * ne + 5 * npp + 4 * no,
                npp)


    check("T13-01", "carbon per unit is 18 + 2n in the arm number n",
          [18 + 2 * n for n in range(5)] == [18, 20, 22, 24, 26],
          "C = %s" % [18 + 2 * n for n in range(5)])
    check("T13-02", "carbon count is strictly monotone, so n is recoverable",
          all(18 + 2 * n < 18 + 2 * (n + 1) for n in range(4)),
          "strictly increasing")
    check("T13-03", "the di-(hydroxyethoxy) unit is C22H20O4",
          formula(0, 2, 0) == (22, 20, 4, 0), "%s" % (formula(0, 2, 0),))
    check("T13-04", "the tetra-(hydroxyethoxy) unit is C26H28O8",
          formula(0, 4, 0) == (26, 28, 8, 0), "%s" % (formula(0, 4, 0),))
    check("T13-05", "the tetra-ol is C18H12O4 (parent chrysene plus four OH)"
        ,
          formula(4, 0, 0) == (18, 12, 4, 0), "%s" % (formula(4, 0, 0),))
    ok = True
    for npp in range(5):
        a, b = formula(0, 4 - npp, npp), formula(0, 4, 0)
        ok = ok and tuple(x - y for x, y in zip(a, b)) == (0, npp, 3 * npp,
            npp)
    check("T13-06", "each phosphorylation adds exactly H1O3P1 and no carbon",
          ok, "checked for 0..4 phosphorylations")
    ok = True
    for no in range(5):
        a, b = formula(0, 4 - no, 0, no), formula(0, 4, 0, 0)
        ok = ok and tuple(y - x for x, y in zip(a, b))[:3] == (0, 4 * no, -2
            * no)
    check("T13-07", "reduction of an oxalate arm to an e arm adds H4, "
                    "removes O2, and moves no carbon",
          ok, "checked for 0..4 oxalate arms")
    check("T13-08", "the 6,12-bis(oxalate) acid is C22H12O8",
          formula(0, 0, 0, 2) == (22, 12, 8, 0), "%s" % (formula(0, 0, 0, 2)
              ,))
```

```
carbon_cycle = RD10 * 22 + RT * 26
lipids = F(nu_lip * L) / 2 + ellICL
check("T13-09", "cycle carbon 22 R_D + 26 R_T = 937 196",
      carbon_cycle == 937196, "carbon = %s" % carbon_cycle)
check("T13-10", "route-A lipids are nu L / 2 + ellICL = 6 677, "
                "with four-carbon coproduct 26 708",
      lipids == 6677 and 4 * lipids == 26708,
      "lipids = %s, coproduct C = %s" % (lipids, 4 * lipids))

# ======================================================================
block("T14", "Column impurity: exact sums against the Perron limit",
      "FLOATING limit")
# ======================================================================
K, Kp, Kpp = F(1), F(1, 5), F(1, 10)
cA, cB = F(9, 10), F(1, 100)
Kmat = {("A", "A"): K, ("A", "B"): Kp, ("B", "A"): Kp, ("B", "B"): Kpp}
cc = {"A": cA, "B": cB}
letters = ["A", "B"]
M = [[Kmat[(x, y)] * cc[y] for y in letters] for x in letters]


def vecmat(v, X):
    return [sum(v[k] * X[k][j] for k in range(2)) for j in range(2)]


def b_fraction(ell):
    u = [cA, cB]
    fwd = [u]
    for _ in range(ell - 1):
        fwd.append(vecmat(fwd[-1], M))
    bwd = [[F(1), F(1)]]
    for _ in range(ell - 1):
        prev = bwd[-1]
        bwd.append([sum(M[i][k] * prev[k] for k in range(2)) for i in
            range(2)])
    bwd = bwd[::-1]
    Z = sum(fwd[-1][j] for j in range(2))
    Bsum = sum(fwd[i][1] * bwd[i][1] for i in range(ell))
    return Bsum / (ell * Z)


a11, a22 = float(K * cA), float(Kpp * cB)
eps_off = float(Kp) * math.sqrt(float(cA * cB))
lam_p = 0.5 * (a11 + a22) + math.sqrt(0.25 * (a11 - a22) ** 2 + eps_off
    ** 2)
tt = eps_off / (lam_p - a22)
piB = tt * tt / (1 + tt * tt)
bnd = float(Kp ** 2 * cA * cB / (K * cA - Kpp * cB) ** 2)

check("T14-01", "the Perron limit lies below the stated closed-form bound
    ",
      piB <= bnd, "piB = %.6e <= bound = %.6e" % (piB, bnd))
fb200, fb400 = float(b_fraction(200)), float(b_fraction(400))
rich = 2 * fb400 - fb200
check("T14-02", "Richardson extrapolation of the exact sums matches the "
                "Perron limit within 0.5%",
      abs(rich - piB) / piB < 5e-3,
```

```
        "Richardson = %.6e, Perron = %.6e, rel = %.2e"
        % (rich, piB, abs(rich - piB) / piB))
check("T14-03", "the exact finite-column fraction decreases toward the
    limit",
        fb400 < fb200, "f(200) = %.6e > f(400) = %.6e" % (fb200, fb400))
naive = float(Kp * cB / (K * cA))
check("T14-04", "the naive single-contact estimate overstates the exact "
                "fraction by about fivefold",
        naive > 4 * fb200, "naive = %.3e vs exact = %.3e" % (naive, fb200))
check("T14-05", "the impurity fraction stays below 1e-3 at this
    composition",
        fb200 < 1e-3, "f(200) = %.3e" % fb200)

# ======================================================================
block("T15", "Scission certificates, residence, registry",
        "part FLOATING")
# ======================================================================
pAdm = pA[2] + pA[3] + pA[4]
pWaste = pA[0] + pA[1]
check("T15-01", "admissible and waste masses partition the exit
    distribution",
        pAdm + pWaste == 1, "p_adm = %.6f, p_waste = %.3e"
        % (float(pAdm), float(pWaste)))
check("T15-02", "the admissible mass exceeds 0.998",
        pAdm > F(998, 1000), "p_adm = %.6f" % float(pAdm))
check("T15-03", "the waste class is dynamically isolated: p_0 + p_1 < 1.2
    e-3",
        pWaste < F(12, 10000), "p_waste = %.3e" % float(pWaste))

# M-matrix certificate for the open source
aa = [F(5)] * 5
dd = [F(1, 20)] * 5
uu = [F(1, 100)] * 5
kk = kA + [F(0)]
states = [(n, "b") for n in range(5)] + [(n, ph) for n in (2, 3, 4) for
    ph in "sc"]
idx = {st: i for i, st in enumerate(states)}
nS = len(states)
G = [[F(0)] * nS for _ in range(nS)]
for n in range(5):
    i = idx[(n, "b")]
    if n < 4:
        G[i][idx[(n + 1, "b")]] += kk[n]
    G[i][i] -= kk[n] + e[n]
    if n >= 2:
        G[i][idx[(n, "s")]] += e[n]
        js, jc = idx[(n, "s")], idx[(n, "c")]
        G[js][jc] += aa[n]
        G[js][js] -= aa[n] + uu[n]
        G[jc][js] += dd[n]
        G[jc][jc] -= dd[n] + uu[n]
negG = [[-x for x in row] for row in G]
tau = solve(negG, [F(1)] * nS)
check("T15-04", "the open source is a nonsingular M-matrix: all
    absorption "
                "times are strictly positive",
        all(x > 0 for x in tau),
```

```
          "min tau = %.4g, max tau = %.4g" % (float(min(tau)), float(max(tau)
              )))
    check("T15-05", "the slowest absorption time is finite and near 101",
          100 < max(tau) < 102, "max tau = %.3f" % float(max(tau)))

591 # metastable residence, exact
    r_ret, a_ret, d_ret = F(1, 1000), F(5), F(1, 20)
    sol = solve([[F(1), -a_ret / (r_ret + a_ret)], [F(-1), F(1)]],
                [1 / (r_ret + a_ret), 1 / d_ret])
    Ts_closed = (1 + a_ret / d_ret) / r_ret
596 check("T15-06", "metastable residence: linear solve matches the closed
        form",
          sol[0] == Ts_closed,
          "T_sigma = %s = %.4g" % (Ts_closed, float(Ts_closed)))
    check("T15-07", "the residence amplification is 1 + a/d = 101",
          1 + a_ret / d_ret == 101, "amplification = %s" % (1 + a_ret / d_ret
              ))
601
    # registry suppression (floating)
    supp = {}
    for nu_r in (2, 4):
        for Dreg in (1.0, 2.0):
606         supp[(nu_r, Dreg)] = math.exp(-2 * nu_r * Dreg)
    check("T15-08", "the tetrasubstituted stream is protected by twice the "
                    "exponent of the disubstituted",
          abs(supp[(4, 1.0)] - supp[(2, 1.0)] ** 2) < 1e-15,
          "s(nu=4) = %.3e = s(nu=2)^2 = %.3e"
611       % (supp[(4, 1.0)], supp[(2, 1.0)] ** 2))
    check("T15-09", "suppression strengthens with the registry free energy",
          supp[(2, 2.0)] < supp[(2, 1.0)] and supp[(4, 2.0)] < supp[(4, 1.0)
              ],
          "s(2,1) = %.3e > s(2,2) = %.3e" % (supp[(2, 1.0)], supp[(2, 2.0)]))
    check("T15-10", "the registry ratio between the two streams is "
616                 "exp(-4) approx 0.0183",
          abs(supp[(4, 1.0)] / supp[(2, 1.0)] - math.exp(-4)) < 1e-12,
          "ratio = %.4f" % (supp[(4, 1.0)] / supp[(2, 1.0)]))
    check("T15-11", "registry lowers the column impurity bound below 1e-5 "
                    "for the disubstituted stream at Delta = 1",
621       bnd * supp[(2, 1.0)] < 1e-5,
          "bound*s = %.3e" % (bnd * supp[(2, 1.0)]))

    # ======================================================================
    print()
626 print("=" * 72)
    print("  RESULT  %d/%d passed,  %d failed" % (PASSED, PASSED + FAILED,
        FAILED))
    print("=" * 72)
    sys.exit(1 if FAILED else 0)
```

Console Transcript: Default Run

$ python3 verify_A79.py

```
======================================================================
  verify_A79.py -- Article 79, 'The Source'
  Annals of the Chrysene Formalism, Vol. 1, Issue 5
======================================================================

======================================================================
  T1  Imported demand
======================================================================
[PASS] T1-01     R_D base at alpha=1/10 is 42 200 units (Art. 65 Thm 8.5)
[PASS] T1-02     R_D at alpha=1/10 adds 2*ellICL: 42 354
[PASS] T1-03     R_D at alpha=2/100 is 136478/7
[PASS] T1-04     R_T founder draw is L + nICL = 208
[PASS] T1-05     demand ratio R_D/R_T reduces to 1629/8
[PASS] T1-06     R_D/R_T rounds to 204 (2*|R_D - 204 R_T| < R_T)
[PASS] T1-07     R_D(0.02)/R_T rounds to 94
[PASS] T1-08     the D demand exceeds the founder pore draw 100-fold

======================================================================
  T2  The arm ladder and the Ordering theorem (Thm 5.1)
======================================================================
[PASS] T2-01     H_add gives a geometric ladder e_n = e_0 q^n, q = 10
[PASS] T2-02     q exceeds unity, so every arm raises the export rate
[PASS] T2-03     Ordering (Thm 5.1): P_0 < P_1 < P_2 < P_3 < P_4
[PASS] T2-04     the ladder spans q^4 = 10 000 from P_0 to P_4
[PASS] T2-05     e_2 = 1 fixes the time unit

======================================================================
  T3  Exit distribution under the adopted kinetics
======================================================================
[PASS] T3-01     closed form agrees with the fundamental matrix exactly
[PASS] T3-02     the exit distribution sums to one exactly
[PASS] T3-03     every exit probability is strictly positive
[PASS] T3-04     the disubstituted channel carries the mass: p_2 > 0.99
[PASS] T3-05     p_3 / p_4 = 1/q: the trisubstituted channel is a tenth
[PASS] T3-06     mean residence is finite and near one time unit

======================================================================
  T4  The design equation (Thm 7.3)
======================================================================
[PASS] T4-01     p_2 / p_4 equals the demand ratio R_D / R_T exactly
[PASS] T4-02     cross-multiplied: p_2 R_T = p_4 R_D
[PASS] T4-03     k_3 solves the design equation
[PASS] T4-04     k_3 is far below k_1, k_2, k_4: the third arm is the
   gate

======================================================================
  T5  Matched efficiency and its lower bound
======================================================================
[PASS] T5-01     matched efficiency eta = p_2 + p_4
[PASS] T5-02     eta meets the stated lower bound
[PASS] T5-03     eta lies strictly below one: the source is not perfect
[PASS] T5-04     eta exceeds 0.998
```

```
[PASS] T5-05     counting the initiator as delivered raises efficiency by
    exactly c_I / Phi
[PASS] T5-06     throughput Phi = R_D / p_2 is near 42 630

======================================================================
  T6  Cooperativity trade-off
======================================================================
[PASS] T6-01     k_4 / k_3 = 203625/11, floor 18 511
[PASS] T6-02     at completion ratio e_3/k_4 = 1/10 the match holds and
   k_4/k_3 = q(R_D/R_T)/(x(1+x))
[PASS] T6-03     at completion ratio e_3/k_4 = 1 the match holds and k_4/
   k_3 = q(R_D/R_T)/(x(1+x))
[PASS] T6-04     at completion ratio e_3/k_4 = 10 the match holds and k_4
   /k_3 = q(R_D/R_T)/(x(1+x))
[PASS] T6-05     the cooperativity requirement falls as completion slows
[PASS] T6-06     the I-loss term grows with the completion ratio

======================================================================
  T7  Initiator surplus
======================================================================
[PASS] T7-01     initiator made per period is 104/5 = 20.8
[PASS] T7-02     the initiator drawn does not exceed the initiator made
[PASS] T7-03     the surplus is at least twentyfold

======================================================================
  T8  Contrast case: slow completion
======================================================================
[PASS] T8-01     the contrast distribution also sums to one
[PASS] T8-02     p_2 falls to about 0.90 when completion is slow
[PASS] T8-03     p_4 collapses below 1e-6
[PASS] T8-04     p_4 / p_3 = k_4 / e_3 under slow completion
[PASS] T8-05     axis window threshold (1-eps)^2/(q eps^2) = 81/10 at eps
   =0.1
[PASS] T8-06     completion threshold q(1-eps)/eps = 90 at eps=0.1

======================================================================
  T9  Throughput and efficiency
======================================================================
[PASS] T9-01     the contrast case needs a vastly larger throughput
[PASS] T9-02     contrast efficiency collapses below 1e-4
[PASS] T9-03     adopted efficiency exceeds the contrast by over four
   decades
[PASS] T9-04     the adopted throughput is finite and near 42 630

======================================================================
  T10  Two-zone variant: Bateman crossing   [FLOATING]
======================================================================
[PASS] T10-01    x* solves x = 2(1 - e^-x), x* approx 1.5936
[PASS] T10-02    x* is the unique positive root: the residual vanishes
[PASS] T10-03    r*delta converges to x* as r grows
[PASS] T10-04    the convergence is first order: error*r is bounded

======================================================================
  T11  Arm words within Article 77's families
======================================================================
[PASS] T11-01    Burnside counts agree with brute force at m=2
[PASS] T11-02    Burnside counts agree with brute force at m=3
```

```
  [PASS] T11-03    Burnside counts agree with brute force at m=4
  [PASS] T11-04    two arm types give 21 words: 3 + 8 + 10
  [PASS] T11-05    three arm types give 78 words: 6 + 27 + 45
  [PASS] T11-06    the tetrasubstituted count is the Burnside average

  ========================================================================
    T12  Arm-efficiency sensitivity
  ========================================================================
  [PASS] T12-01    arm efficiency at per-arm fidelity 0.99 is below one and
      falls with fidelity
  [PASS] T12-02    arm efficiency at per-arm fidelity 0.95 is below one and
      falls with fidelity
  [PASS] T12-03    arm efficiency at per-arm fidelity 0.90 is below one and
      falls with fidelity
  [PASS] T12-04    even at fidelity 0.90 the arm efficiency stays above
     0.98

  ========================================================================
    T13  Carbon per admitted unit and molecular formulae
  ========================================================================
  [PASS] T13-01    carbon per unit is 18 + 2n in the arm number n
  [PASS] T13-02    carbon count is strictly monotone, so n is recoverable
  [PASS] T13-03    the di-(hydroxyethoxy) unit is C22H2004
  [PASS] T13-04    the tetra-(hydroxyethoxy) unit is C26H2808
  [PASS] T13-05    the tetra-ol is C18H1204 (parent chrysene plus four OH)
  [PASS] T13-06    each phosphorylation adds exactly H103P1 and no carbon
  [PASS] T13-07    reduction of an oxalate arm to an e arm adds H4, removes
      O2, and moves no carbon
  [PASS] T13-08    the 6,12-bis(oxalate) acid is C22H1208
  [PASS] T13-09    cycle carbon 22 R_D + 26 R_T = 937 196
  [PASS] T13-10    route-A lipids are nu L / 2 + ellICL = 6 677, with four-
     carbon coproduct 26 708

  ========================================================================
    T14  Column impurity: exact sums against the Perron limit   [FLOATING
       limit]
  ========================================================================
  [PASS] T14-01    the Perron limit lies below the stated closed-form bound
  [PASS] T14-02    Richardson extrapolation of the exact sums matches the
     Perron limit within 0.5%
  [PASS] T14-03    the exact finite-column fraction decreases toward the
     limit
  [PASS] T14-04    the naive single-contact estimate overstates the exact
     fraction by about fivefold
  [PASS] T14-05    the impurity fraction stays below 1e-3 at this
     composition

  ========================================================================
    T15  Scission certificates, residence, registry   [part FLOATING]
  ========================================================================
  [PASS] T15-01    admissible and waste masses partition the exit
     distribution
  [PASS] T15-02    the admissible mass exceeds 0.998
  [PASS] T15-03    the waste class is dynamically isolated: p_0 + p_1 < 1.2
     e-3
  [PASS] T15-04    the open source is a nonsingular M-matrix: all
     absorption times are strictly positive
```

```
[PASS] T15-05    the slowest absorption time is finite and near 101
[PASS] T15-06    metastable residence: linear solve matches the closed
    form
[PASS] T15-07    the residence amplification is 1 + a/d = 101
[PASS] T15-08    the tetrasubstituted stream is protected by twice the
    exponent of the disubstituted
[PASS] T15-09    suppression strengthens with the registry free energy
[PASS] T15-10    the registry ratio between the two streams is exp(-4)
    approx 0.0183
[PASS] T15-11    registry lowers the column impurity bound below 1e-5 for
     the disubstituted stream at Delta = 1

=========================================================================
  RESULT  88/88 passed,  0 failed
=========================================================================
```

Console Transcript: Verbose Run (-v)

$ python3 verify_A79.py -v

```
======================================================================
  verify_A79.py -- Article 79, 'The Source'
  Annals of the Chrysene Formalism, Vol. 1, Issue 5
======================================================================

======================================================================
  T1  Imported demand
======================================================================
[PASS] T1-01     R_D base at alpha=1/10 is 42 200 units (Art. 65 Thm 8.5)
    R_D_base = 42200
[PASS] T1-02     R_D at alpha=1/10 adds 2*ellICL: 42 354
    R_D = 42354 = 42200 + 2*77
[PASS] T1-03     R_D at alpha=2/100 is 136478/7
    R_D(0.02) = 136478/7
[PASS] T1-04     R_T founder draw is L + nICL = 208
    R_T = 208
[PASS] T1-05     demand ratio R_D/R_T reduces to 1629/8
    R_D/R_T = 1629/8, gcd = 26
[PASS] T1-06     R_D/R_T rounds to 204 (2*|R_D - 204 R_T| < R_T)
    2*78 = 156 < 208
[PASS] T1-07     R_D(0.02)/R_T rounds to 94
    2*386 < 1456
[PASS] T1-08     the D demand exceeds the founder pore draw 100-fold
    100 R_T = 20800 < R_D = 42354

======================================================================
  T2  The arm ladder and the Ordering theorem (Thm 5.1)
======================================================================
[PASS] T2-01     H_add gives a geometric ladder e_n = e_0 q^n, q = 10
    e = ['1/100', '1/10', '1', '10', '100']
[PASS] T2-02     q exceeds unity, so every arm raises the export rate
    q = 10
[PASS] T2-03     Ordering (Thm 5.1): P_0 < P_1 < P_2 < P_3 < P_4
    strictly increasing across all four steps
[PASS] T2-04     the ladder spans q^4 = 10 000 from P_0 to P_4
    P_4/P_0 = 10000
[PASS] T2-05     e_2 = 1 fixes the time unit
    e_2 = 1

======================================================================
  T3  Exit distribution under the adopted kinetics
======================================================================
[PASS] T3-01     closed form agrees with the fundamental matrix exactly
    max difference = 0
[PASS] T3-02     the exit distribution sums to one exactly
    sum = 1
[PASS] T3-03     every exit probability is strictly positive
    min = 1/10001
[PASS] T3-04     the disubstituted channel carries the mass: p_2 > 0.99
    p_2 = 0.993534
[PASS] T3-05     p_3 / p_4 = 1/q: the trisubstituted channel is a tenth
    p_3/p_4 = 1/10
```

```
  [PASS] T3-06     mean residence is finite and near one time unit
54       mean residence = 1.014

  ============================================================================
    T4  The design equation (Thm 7.3)
  ============================================================================
59 [PASS] T4-01     p_2 / p_4 equals the demand ratio R_D / R_T exactly
       p_2/p_4 = 1629/8 = R_D/R_T
   [PASS] T4-02     cross-multiplied: p_2 R_T = p_4 R_D
       both = 1303200000000/6306160553
   [PASS] T4-03     k_3 solves the design equation
64     k_3 = 44/8145 = 0.00540209
   [PASS] T4-04     k_3 is far below k_1, k_2, k_4: the third arm is the
      gate
       k_3 = 0.00540209, k_1 = 100

  ============================================================================
69   T5  Matched efficiency and its lower bound
  ============================================================================
   [PASS] T5-01     matched efficiency eta = p_2 + p_4
       eta = 0.998413
   [PASS] T5-02     eta meets the stated lower bound
74     eta = 0.998413 >= bound = 0.998411
   [PASS] T5-03     eta lies strictly below one: the source is not perfect
       1 - eta = 1.587e-03
   [PASS] T5-04     eta exceeds 0.998
       eta = 0.998413
79 [PASS] T5-05     counting the initiator as delivered raises efficiency by
       exactly c_I / Phi
       eta_full - eta = 2.346e-05 = c_I/Phi = 2.346e-05
   [PASS] T5-06     throughput Phi = R_D / p_2 is near 42 630
       Phi = 42629.65

84 ============================================================================
    T6  Cooperativity trade-off
  ============================================================================
   [PASS] T6-01     k_4 / k_3 = 203625/11, floor 18 511
       k_4/k_3 = 203625/11 = 18511.4
89 [PASS] T6-02     at completion ratio e_3/k_4 = 1/10 the match holds and
      k_4/k_3 = q(R_D/R_T)/(x(1+x))
       k_4/k_3 = 1.851e+04, p_3 = 0.0488%
   [PASS] T6-03     at completion ratio e_3/k_4 = 1 the match holds and k_4/
      k_3 = q(R_D/R_T)/(x(1+x))
       k_4/k_3 = 1018, p_3 = 0.4858%
   [PASS] T6-04     at completion ratio e_3/k_4 = 10 the match holds and k_4
      /k_3 = q(R_D/R_T)/(x(1+x))
94     k_4/k_3 = 18.51, p_3 = 4.6542%
   [PASS] T6-05     the cooperativity requirement falls as completion slows
       x=0.1 needs 1.851e+04, x=10 needs 18.51
   [PASS] T6-06     the I-loss term grows with the completion ratio
       loss(x=0.1) = 0.0489%, loss(x=10) = 4.8870%
99
  ============================================================================
    T7  Initiator surplus
  ============================================================================
   [PASS] T7-01     initiator made per period is 104/5 = 20.8
104    Phi p_3 = 104/5 = 20.8
```

```
  [PASS] T7-02     the initiator drawn does not exceed the initiator made
      c_I = 1 <= 20.8
  [PASS] T7-03     the surplus is at least twentyfold
      ratio = 20.8

  ========================================================================
    T8  Contrast case: slow completion
  ========================================================================
  [PASS] T8-01     the contrast distribution also sums to one
      sum = 1
  [PASS] T8-02     p_2 falls to about 0.90 when completion is slow
      p_2 = 0.903
  [PASS] T8-03     p_4 collapses below 1e-6
      p_4 = 1.805e-07
  [PASS] T8-04     p_4 / p_3 = k_4 / e_3 under slow completion
      ratio = 1.00e-04
  [PASS] T8-05     axis window threshold (1-eps)^2/(q eps^2) = 81/10 at eps
     =0.1
      threshold = 81/10 = 8.10
  [PASS] T8-06     completion threshold q(1-eps)/eps = 90 at eps=0.1
      threshold = 90

  ========================================================================
    T9  Throughput and efficiency
  ========================================================================
  [PASS] T9-01     the contrast case needs a vastly larger throughput
      Phi_B/Phi_A = 2.7e+04
  [PASS] T9-02     contrast efficiency collapses below 1e-4
      eta_B = 3.69e-05
  [PASS] T9-03     adopted efficiency exceeds the contrast by over four
     decades
      ratio = 2.7e+04
  [PASS] T9-04     the adopted throughput is finite and near 42 630
      Phi_A = 42629.65

  ========================================================================
    T10  Two-zone variant: Bateman crossing   [FLOATING]
  ========================================================================
  [PASS] T10-01    x* solves x = 2(1 - e^-x), x* approx 1.5936
      x* = 1.593624
  [PASS] T10-02    x* is the unique positive root: the residual vanishes
      residual = 0.000e+00
  [PASS] T10-03    r*delta converges to x* as r grows
      r=10000: r*delta = 1.59344 vs x* = 1.59362
  [PASS] T10-04    the convergence is first order: error*r is bounded
      max error*r = 1.815

  ========================================================================
    T11  Arm words within Article 77's families
  ========================================================================
  [PASS] T11-01    Burnside counts agree with brute force at m=2
      closed = (3, 8, 10), brute = (3, 8, 10)
  [PASS] T11-02    Burnside counts agree with brute force at m=3
      closed = (6, 27, 45), brute = (6, 27, 45)
  [PASS] T11-03    Burnside counts agree with brute force at m=4
      closed = (10, 64, 136), brute = (10, 64, 136)
  [PASS] T11-04    two arm types give 21 words: 3 + 8 + 10
```

```
      3 + 8 + 10 = 21
  [PASS] T11-05    three arm types give 78 words: 6 + 27 + 45
      6 + 27 + 45 = 78
  [PASS] T11-06    the tetrasubstituted count is the Burnside average
164     2T(m) = m^4 + m^2 for m = 2,3,4

  ============================================================================
    T12  Arm-efficiency sensitivity
  ============================================================================
169 [PASS] T12-01    arm efficiency at per-arm fidelity 0.99 is below one and
       falls with fidelity
      eta_arm = 0.9997
  [PASS] T12-02    arm efficiency at per-arm fidelity 0.95 is below one and
       falls with fidelity
      eta_arm = 0.9964
  [PASS] T12-03    arm efficiency at per-arm fidelity 0.90 is below one and
       falls with fidelity
174     eta_arm = 0.9875
  [PASS] T12-04    even at fidelity 0.90 the arm efficiency stays above
      0.98
      eta_arm(0.90) = 0.9875

  ============================================================================
179   T13  Carbon per admitted unit and molecular formulae
  ============================================================================
  [PASS] T13-01    carbon per unit is 18 + 2n in the arm number n
      C = [18, 20, 22, 24, 26]
  [PASS] T13-02    carbon count is strictly monotone, so n is recoverable
184     strictly increasing
  [PASS] T13-03    the di-(hydroxyethoxy) unit is C22H20O4
      (22, 20, 4, 0)
  [PASS] T13-04    the tetra-(hydroxyethoxy) unit is C26H28O8
      (26, 28, 8, 0)
189 [PASS] T13-05    the tetra-ol is C18H12O4 (parent chrysene plus four OH)
      (18, 12, 4, 0)
  [PASS] T13-06    each phosphorylation adds exactly H1O3P1 and no carbon
      checked for 0..4 phosphorylations
  [PASS] T13-07    reduction of an oxalate arm to an e arm adds H4, removes
       O2, and moves no carbon
194     checked for 0..4 oxalate arms
  [PASS] T13-08    the 6,12-bis(oxalate) acid is C22H12O8
      (22, 12, 8, 0)
  [PASS] T13-09    cycle carbon 22 R_D + 26 R_T = 937 196
      carbon = 937196
199 [PASS] T13-10    route-A lipids are nu L / 2 + ellICL = 6 677, with four-
      carbon coproduct 26 708
      lipids = 6677, coproduct C = 26708

  ============================================================================
    T14  Column impurity: exact sums against the Perron limit   [FLOATING
       limit]
204 ============================================================================
  [PASS] T14-01    the Perron limit lies below the stated closed-form bound
      piB = 4.448394e-04 <= bound = 4.454337e-04
  [PASS] T14-02    Richardson extrapolation of the exact sums matches the
      Perron limit within 0.5%
      Richardson = 4.448394e-04, Perron = 4.448394e-04, rel = 0.00e+00
```

```
[PASS] T14-03    the exact finite-column fraction decreases toward the limit
    f(200) = 4.625905e-04 > f(400) = 4.537150e-04
[PASS] T14-04    the naive single-contact estimate overstates the exact fraction by about fivefold
    naive = 2.222e-03 vs exact = 4.626e-04
[PASS] T14-05    the impurity fraction stays below 1e-3 at this composition
    f(200) = 4.626e-04

========================================================================
  T15  Scission certificates, residence, registry   [part FLOATING]
========================================================================
[PASS] T15-01    admissible and waste masses partition the exit distribution
    p_adm = 0.998901, p_waste = 1.099e-03
[PASS] T15-02    the admissible mass exceeds 0.998
    p_adm = 0.998901
[PASS] T15-03    the waste class is dynamically isolated: p_0 + p_1 < 1.2e-3
    p_waste = 1.099e-03
[PASS] T15-04    the open source is a nonsingular M-matrix: all absorption times are strictly positive
    min tau = 100, max tau = 101
[PASS] T15-05    the slowest absorption time is finite and near 101
    max tau = 100.995
[PASS] T15-06    metastable residence: linear solve matches the closed form
    T_sigma = 101000 = 1.01e+05
[PASS] T15-07    the residence amplification is 1 + a/d = 101
    amplification = 101
[PASS] T15-08    the tetrasubstituted stream is protected by twice the exponent of the disubstituted
    s(nu=4) = 3.355e-04 = s(nu=2)^2 = 3.355e-04
[PASS] T15-09    suppression strengthens with the registry free energy
    s(2,1) = 1.832e-02 > s(2,2) = 3.355e-04
[PASS] T15-10    the registry ratio between the two streams is exp(-4) approx 0.0183
    ratio = 0.0183
[PASS] T15-11    registry lowers the column impurity bound below 1e-5 for the disubstituted stream at Delta = 1
    bound*s = 8.158e-06

========================================================================
  RESULT  88/88 passed,  0 failed
========================================================================
```

Lean 4 Proof Summary: `acf-79-proof.lean`

Lean Kernel Certification

The file `acf-79-proof.lean` contains 55 theorems in the namespace `ChryseneFormalism.Article79`, checked by the Lean 4.12.0 kernel. Every theorem is discharged by `decide`: the goal is a decidable proposition over `Nat` and the kernel evaluates a verified Boolean decision procedure. No Mathlib, no `sorry`, no `native_decide`.
An `#print axioms` audit over all 55 theorems returns *"does not depend on any axioms"* for 54 of them; the single exception, `demand_gcd`, depends on `propext` alone, which it inherits from `Nat.gcd` in the Lean core. Neither `Quot.sound` nor `funext` nor `Classical.choice` is used anywhere in the file. The full dossier for the proof is published separately as `acf-79-proof.tex`.

Relation between the two artefacts

The Python harness and the Lean file overlap deliberately but are not redundant. The harness covers everything, including the floating-point blocks, and is the broader instrument. The Lean file covers only the finite arithmetic, and covers it to a higher standard: a `decide` proof is checked by the kernel rather than by an interpreter, and a disagreement between the two would be a defect in one of them. Where a claim appears in both, the Lean statement is the stronger certificate.