# **Doc 4A — Algorithm: Step Order, Tabulation & Denominators (rewritten)**

**Scope of this part:** mandatory run **step order (1–8)**, ballot tabulation rules, blank/invalid handling, **denominator policy** (incl. approval gate), and **PR entry threshold**.  
 **Variables referenced:** **VM-VAR-001..007, 012** (Doc 2A).  
 **Rounding policy:** **round half to even** for internal comparisons; **percentages rounded only at reporting** (Doc 7).

## **1) Mandatory step order (must be followed exactly)**

1. **VALIDATE inputs** — structural & semantic checks (tree, magnitudes, tallies sanity, required data present).
2. **TABULATE ballots** per Unit — produce per-option **UnitScores** according to **VM-VAR-001**.
3. **ALLOCATE seats/power** per Unit — apply allocation method (Doc 4B) using UnitScores.
4. **AGGREGATE** to parent levels — roll Unit allocations using weighting (Doc 4B).
5. **APPLY DECISION GATES** — quorum → majority/supermajority → double-majority → symmetry (Doc 4C).
6. **MAP FRONTIER** *(if enabled)* — translate support to status and check contiguity (Doc 4C).
7. **RESOLVE TIES** *(only if blocking)* — per tie policy (Doc 4C).
8. **LABEL & PACKAGE** — assign Decisive/Marginal/Invalid and assemble Result.

*(If validation fails: skip 2–7 and still package an* ***Invalid*** *Result with reasons. If gates fail: skip frontier.)*

## **2) Ballot tabulation rules (how UnitScores are computed)**

### **2.1 Plurality — VM-VAR-001 = plurality**

* Each ballot selects **one** Option.
* **Unit score** for an Option = raw count of ballots selecting it.
* **Support % (for gates where applicable):** votes\_for\_change / valid\_ballots.
* Blank/invalid: excluded from valid ballot count (see §3).

### **2.2 Approval — VM-VAR-001 = approval**

* Each ballot may approve **any number** of Options.
* **Unit score** for an Option = total approvals it receives.
* **Allocation math** (e.g., PR) uses these approval *counts*.
* **Decision-gate support % is fixed to the *approval rate*:** **approval\_rate(change) = approvals\_for\_change / valid\_ballots**. *(This is mandatory, not configurable.)*
* Blank/invalid: excluded from valid ballot count.

### **2.3 Score — VM-VAR-001 = score, VM-VAR-002/003/004**

* Each ballot scores each Option on the fixed scale **[VM-VAR-002 … VM-VAR-003]**.
* If **VM-VAR-004 = linear**, normalize each ballot to the scale span before summing; otherwise use raw scores.
* **Unit score** for an Option = **sum of (possibly normalized) scores**.
* **Support % for gates (binary change vs status quo):** score\_sum\_for\_change / (maximum\_possible\_score\_per\_ballot × valid\_ballots) — *only if a binary gate is applied; otherwise gates validate election integrity (e.g., quorum) rather than choose a winner.*
* Mean scores may be reported, but do **not** drive allocation.

### **2.4 Ranked — IRV — VM-VAR-001 = ranked\_irv, VM-VAR-006**

* Repeatedly **eliminate** the lowest-tally Option; **transfer** each ballot to its next **continuing** preference.
* Stop when an Option reaches a **majority of continuing ballots**, or only one Option remains.
* **Exhausted ballots policy (fixed):** **VM-VAR-006 = reduce\_continuing\_denominator** — once a ballot has no further ranked options, it is **excluded from the continuing-ballots denominator** for subsequent rounds.
* Round logs (eliminations, transfers, exhausted counts) are produced for audit.

### **2.5 Ranked — Condorcet — VM-VAR-001 = ranked\_condorcet, VM-VAR-005**

* Compute **pairwise** contests among all Options using valid rankings.
* If a **Condorcet winner** (beats each other Option head-to-head) exists, it wins the Unit.
* If not, apply **VM-VAR-005** (schulze or minimax) as the **completion rule** to select the Unit winner.
* Pairwise matrix is produced for audit.

## **3) Blank/invalid ballots (common to all ballot types)**

* **Count toward turnout** (ballots\_cast).
* **Do not** contribute to any Option’s score.
* **Default denominator for support %** uses **valid ballots (excludes blank/invalid)**.
* If **VM-VAR-007 = on** (include\_blank\_in\_denominator), **blank/invalid are included** in the denominator **for majority/supermajority gates only**; tabulation and allocation still use valid ballots/approvals/scores as defined above.

## **4) PR entry threshold (unit-level eligibility)**

* **VM-VAR-012 pr\_entry\_threshold\_pct** (integer %) applies **only** to proportional allocation methods.
* Per Unit, any Option with share **below** the threshold is **ineligible for seats in that Unit**.
* The share is computed using the **natural tabulation denominator** of the ballot type (e.g., approvals share of total approvals for approval-PR, vote share for plurality-PR, score share for score-PR).
* Threshold does **not** alter raw UnitScores; it filters candidates for the **allocation** step.

## **5) Denominator policy (summary)**

* **Tabulation:**
  + Plurality → counts over **valid ballots**.
  + Approval → **counts of approvals** per Option (for allocation).
  + Score → **sum of scores** per Option (for allocation).
  + Ranked (IRV/Condorcet) → method-specific tallies; continuing-ballots denominator shrinks under IRV exhaustion.
* **Decision gates:**
  + **Approval gate is fixed** to **approval rate = approvals\_for\_change / valid ballots** (not approvals share).
  + Other ballot types use **support / valid ballots** unless **VM-VAR-007 = on**, in which case **valid+blank** form the denominator **for gates only**.
* **Reporting:** percentages rounded **once** at presentation; internal math uses exact integers/rationals with **round half to even** at the defined comparison points.

### **Completion checklist (for this part)**

* Step order 1–8 fixed.
* Ballot rules for plurality, approval, score, IRV (exhaustion stated), Condorcet+completion.
* **Approval gate denominator** locked to **approval rate**.
* Blank/invalid handling clarified; **VM-VAR-007** scope limited to gates.
* **PR threshold (VM-VAR-012)** defined and scoped.
* Rounding/percent presentation rules restated.

# **Doc 4B — Algorithm: Allocation & Aggregation (incl. MMP)**

**Scope of this part:** unit-level **allocation math** (WTA, proportional variants, LR), **Mixed Local + Correction (MMP)** sequence with fixed controls, and **aggregation** up the hierarchy.  
 **Variables referenced:** **VM-VAR-010..015, 030..031, 016, 017** (Doc 2A/2C).  
 **Functions implementing this:** **VM-FUN-004 AllocateUnit**, **VM-FUN-005 AggregateHierarchy** (Doc 5).

## **1) Preliminaries (what enters allocation)**

* **Inputs per Unit:**
  + **UnitScores** from tabulation (Doc 4A).
  + **Unit.magnitude** (integer ≥1).
  + **PR entry threshold:** **VM-VAR-012** (applies to proportional/LR only).
* **General tie for last seat:** break by higher raw Unit score; if still tied, use deterministic order; if tie\_policy=random, draw with seed (see Doc 4C).

**Constraint:** If allocation\_method = winner\_take\_all then **Unit.magnitude must be 1**. Otherwise the run is **Invalid** (validated in VM-FUN-002).

## **2) Allocation methods (per Unit)**

### **2.1 Winner-take-all (WTA) — VM-VAR-010 = winner\_take\_all**

* Winner = Option with **highest Unit score**.
* Seats/power: **100%** to the winner (since m=1).
* Ties handled per tie policy.

### **2.2 Proportional — favor big (D’Hondt) — VM-VAR-010 = proportional\_favor\_big**

* Sequential highest-average using divisor sequence: **1, 2, 3, …**
* Repeat until **m** seats assigned: at each step, give the seat to the Option maximizing  
   score / (seats\_already\_assigned + next\_divisor).
* Apply **PR entry threshold** (**VM-VAR-012**) first (exclude below-threshold Options).

### **2.3 Proportional — favor small (Sainte-Laguë) — VM-VAR-010 = proportional\_favor\_small**

* Sequential highest-average using **odd** divisors: **1, 3, 5, …**
* Procedure as in 2.2 with the odd sequence.
* Apply **PR entry threshold** beforehand.

### **2.4 Largest Remainder (LR) — VM-VAR-010 = largest\_remainder**

1. Compute ideal seats per Option: ideal = m × (score / sum\_scores).
2. Assign floor(ideal) to each.
3. Distribute the **remaining seats** to the largest fractional remainders.

* Apply **PR entry threshold** beforehand.

**Note:** “score” means the ballot’s **natural** tally (approvals for approval, votes for plurality, score sums for score).

## **3) Mixed Local + Correction (MMP-style) — VM-VAR-010 = mixed\_local\_correction**

**Purpose:** keep **local representation** (single-member WTA seats) while adding a **correction tier** to align total seat shares to **targets**.

### **3.1 Fixed controls (from variables)**

* **Top-up share (percent of total seats)**: **VM-VAR-013 mlc\_topup\_share\_pct**.
* **Target basis:** **VM-VAR-015 target\_share\_basis = natural\_vote\_share** (v1).
* **Correction level:** **VM-VAR-016 mlc\_correction\_level ∈ {national, regional}**.
* **Total seats model:** **VM-VAR-017 total\_seats\_model ∈ {fixed\_total, variable\_add\_seats}**.
* **Overhang handling:** **VM-VAR-014 ∈ {allow\_overhang, compensate\_others, add\_total\_seats}**.

### **3.2 Seat pools**

Let **B** = sum of **base local seats** (usually one per Unit; or the registry’s local magnitudes).

* If **fixed\_total**: total seats **T** are fixed and known. The **top-up pool** size is  
   TopUp = floor( (VM-VAR-013 / 100) × T ), and \*\*Local = T − TopUp`.
* If **variable\_add\_seats**: start with **T₀ = B** and \*\*TopUp₀ = floor( (VM-VAR-013 / 100) × T₀ ). Seats may \*\*increase\*\* if add\_total\_seats` is chosen (see 3.5).

In both models, **local seats are assigned first** and are never taken away.

### **3.3 Targets**

At the **correction level** (national or each region, per **VM-VAR-016**):

* Compute **vote shares** from the ballot’s **natural totals** (Doc 4A).
* **Target seats** per Option = share × T\_level, where T\_level is:  
  + **fixed\_total:** the level’s portion of **T** (for national, T itself; for regional, sum of seats in that region).
  + **variable\_add\_seats:** at iteration *k*, use current T\_level(k) (see 3.5).

### **3.4 Assign top-up seats**

* For each level, compute **deficit = target − (local seats already won)**.
* Iteratively assign **one top-up seat** at a time to the Option with the **largest positive deficit** (ties: higher vote share, then deterministic order, then random by seed if policy allows), until either:  
  + the **TopUp** pool for that level is exhausted, or
  + all deficits are **≤ 0** (no one under target).

### **3.5 Overhang & total seats model interaction — VM-VAR-014, VM-VAR-017**

* **allow\_overhang (default):** If a party’s **local seats > target**, it keeps them. Others may still receive top-ups from the pool, but no seats are taken away; resulting **totals can exceed targets**.
* **compensate\_others:** Same as allow\_overhang for the overhung party, but **assign remaining top-ups preferentially** to non-overhung parties. **Total seats stay at T** (fixed\_total) or **T₀ + TopUp₀** (variable model with no add).
* **add\_total\_seats (only meaningful with variable\_add\_seats):** If deficits remain after consuming **TopUp₀**, **increase T** by adding seats one by one—assigning each new seat to the **largest remaining deficit**—until all deficits are **≤ 0** or a documented policy cap is hit. Record the final **T** in Result.

**Important:** Overhang never **removes** local seats already won. The correction only **adds** seats.

### **3.6 Outcome**

* Final per-Option seat totals at the correction level = local seats + top-ups (+ any added seats if add\_total\_seats).
* Distribute level totals back down to Units as:  
  + **Local seats:** already known per Unit.
  + **Top-up seats:** reported at the **correction level** (national/regional); they are not bound to specific Units in v1.

## **4) Aggregation (roll-up across hierarchy) — VM-VAR-030, 031**

After each Unit’s allocation:

### **4.1 Weighting method — VM-VAR-030**

* **equal\_unit:** each Unit contributes **equally** at its parent level.
* **population\_baseline:** each Unit is weighted by its **population\_baseline** from the registry (Doc 1), using the provenance year recorded there.

### **4.2 Procedure**

* For each parent level (District→Region→Country):  
  1. **Sum seats** by Option from child Units (**not** the raw scores).
  2. Compute **shares** by Option at that level.
  3. Carry **turnout & validity flags** needed for gates and reporting.
* National decisions are taken at **aggregate\_level = country** (v1; **VM-VAR-031**).

## **5) Cross-references & invariants**

* **Variables:** VM-VAR-010..015 (allocation families & MMP knobs), **013** (top-up share), **016** (correction level), **017** (total seats model), **030..031** (weighting).
* **Functions:** VM-FUN-004 (per-Unit allocation), VM-FUN-005 (aggregation), MMP arithmetic reported in Results (Doc 5).
* **Invariants:**
  + For PR/LR methods, apply **PR entry threshold** (**VM-VAR-012**) **before** seat math.
  + **WTA ⇒ m=1** (else Invalid).
  + Aggregation must use a **stable, total order** on IDs for any list operations (determinism).
  + No rounding to presentation precision until report layer; internal rounding uses **round half to even** only where comparisons require it.

**Done:** Seat math for WTA/PR/LR is precise; **MMP sequencing** (pools, targets, deficits, overhang & total-seats model) is explicit; aggregation by weighting is unambiguous.

# **Doc 4C — Algorithm: Gates, Frontier, Ties, Labels & Edge (rewritten)**

**Scope of this part:** legitimacy gates (quorum/majority/double-majority/symmetry), frontier mapping (binary/sliding/ladder) with contiguity & protections, tie policy, decisiveness labels, and explicit edge cases.  
 **Variables referenced:** **VM-VAR-020..029, 040..048, 050..052, 060..062** (Docs 2A/2C).  
 **Implements:** **VM-FUN-006 ApplyDecisionRules**, **VM-FUN-007 MapFrontier**, **VM-FUN-008 ResolveTies**, **VM-FUN-009 LabelDecisiveness** (Doc 5).

## **1) Decision gates (fixed order)**

### **1.1 Quorum (turnout)**

* **Turnout per country** = sum(ballots\_cast) ÷ sum(eligible\_roll) × 100 (integer % internal).  
  + ballots\_cast from **BallotTally**; eligible\_roll from **Unit** (Doc 1B).
  + The **roll\_inclusion\_policy** (VM-VAR-028) is descriptive; the math always uses eligible\_roll.
* **Global quorum:** **Pass** iff turnout ≥ **VM-VAR-020**.
* **Per-unit quorum:** If **VM-VAR-021 > 0**, a Unit **passes** iff its turnout ≥ **VM-VAR-021**.  
  + **Scope (optional):** **VM-VAR-021\_scope** =  
    - frontier\_only (default): Unit failing per-unit quorum **cannot change status** in frontier mapping, but still counts in any affected-family composition.
    - frontier\_and\_family: Such Units are **excluded** from affected-family calculations in §1.3.

### **1.2 Majority / Supermajority (national)**

* **Required national support** = **VM-VAR-022** (integer %).
* **Denominator for support:**
  + Default = **valid ballots** (excludes blanks/invalid).
  + If **VM-VAR-007 = on**, include blanks/invalid in the **gate denominator** only.
  + **Approval ballots:** **fixed** to **approval rate = approvals\_for\_change / valid ballots** (from Doc 4A).
* **Rule:** **Pass** iff support ≥ threshold (≥, not >).

### **1.3 Double-majority (national + affected-region family)**

* Enabled by **VM-VAR-024 = on**. **Pass** only if **both**:  
  + National support ≥ **VM-VAR-022**, **and**
  + Affected-region family support ≥ **VM-VAR-023** (same denominator policy as §1.2).
* **Affected-region family definition (VM-VAR-026/027):**
  + by\_proposed\_change (default): Units whose status would change under the **current proposal/frontier outcome**.
  + by\_list or by\_tag: explicit linkage from **ParameterSet** / Registry tags.
* **Constraint when no frontier is used:** If **frontier\_mode = none**, by\_proposed\_change is **not allowed**. You **must** use by\_list or by\_tag (validated in VM-FUN-002).
* **Per-unit quorum scope:** If **VM-VAR-021\_scope = frontier\_and\_family**, Units failing per-unit quorum are **excluded** from the family’s support calculation.

### **1.4 Symmetry (threshold neutrality)**

* **VM-VAR-025 = on** requires **identical thresholds and denominators** regardless of direction (A→B or B→A).
* **Exceptions list:** **VM-VAR-029 symmetry\_exceptions** (Units or tagged families with rationale). If non-empty, record **“Not respected”** with the rationale; gates can still pass if substantive thresholds are met.

**Executive elections (note):** Quorum applies as configured. Majority concepts follow the executive ballot logic (e.g., IRV majority of continuing). **Double-majority does not apply** to executives **unless** **VM-VAR-073 = on**.

## **2) Frontier / Autonomy mapping (if VM-VAR-040 ≠ none)**

**Inputs:** Per-Unit support %, **Adjacency** (with type ∈ {land, bridge, water}), contiguity policies (VM-VAR-047/048), protections, and bands/cutoffs (VM-VAR-041/042/046).  
 **Output:** **FrontierMap** with per-Unit status and flags; mediation/enclave/protected overrides are recorded.

### **2.1 Contiguity policy**

* **Edge types allowed to connect Units** = **VM-VAR-047 contiguity\_modes\_allowed** (subset of {land, bridge, water}; default {land, bridge}).
* Build contiguous **components** using only allowed edge types.
* **Island exception (VM-VAR-048):**
  + none (default): Any component not connected to the main area for a given action is flagged **Mediation** (no status change applied in that island).
  + ferry\_allowed: When a component is separated **only by water**, treat **water** edges as **temporarily allowed** for contiguity just for connection to the nearest same-status component; otherwise flag **Mediation**.
  + corridor\_required: **Bridge** edges alone do **not** satisfy contiguity; require a land-only path. Components connected solely via bridges are flagged **Mediation**.

### **2.2 Protected areas**

* Units with protected\_area = true **cannot change status** unless **VM-VAR-045 = on** (protected\_override\_allowed).
* If overridden, mark **protected\_override\_used = true** for those Units in **FrontierMap** and **Result.UnitBlock**.

### **2.3 Per-unit quorum effect on mapping**

* If **VM-VAR-021 > 0** and a Unit’s turnout < per-unit quorum, that Unit **cannot change status** (maps to **no\_change**) regardless of support, and is flagged accordingly. (Family scope impact per §1.1.)

### **2.4 Modes**

**a) Binary cutoff — VM-VAR-040 = binary\_cutoff**

* A Unit changes status iff **support ≥ VM-VAR-041 cutoff\_pct** **and** contiguity is satisfied under §2.1.
* Units in non-contiguous islands (per policy) → **Mediation** (no change).
* Protected rule in §2.2 applies.

**b) Sliding scale — VM-VAR-040 = sliding\_scale**

* Assign each Unit to **exactly one band** from **VM-VAR-042** {min\_pct, max\_pct, action} (non-overlapping; total coverage 0–100).
* Merge adjacent Units with the same action into components per §2.1; flag mediation/enclaves.
* Actions may include no\_change, phased\_change, immediate\_change, or autonomy actions.

**c) Autonomy ladder — VM-VAR-040 = autonomy\_ladder**

* Same banding as sliding scale, but autonomy actions **must map** to **AutonomyPackage IDs** via **VM-VAR-046 autonomy\_package\_map**.
* Record the selected **AP** ID per Unit in **FrontierMap**; reporting references package names/versions.

## **3) Tie resolution (only when a tie blocks a required decision)**

**Policy order (VM-VAR-050..052):**

1. **status\_quo** → Status Quo prevails wherever applicable.
2. **deterministic\_order** → resolve using **Option.order\_index** (lower index wins).
3. **random** → resolve by deterministic RNG with **VM-VAR-052 rng\_seed**; record **TieLog** {context, candidates, policy, seed, winner}.

**Contexts that may require tie resolution:**

* **WTA winner tie** (unit-level).
* **Last seat tie** in proportional/LR allocation.
* **IRV elimination tie** (applies the same policy).
* **Condorcet cycle** is **not** a tie; it is resolved by the **completion rule** (VM-VAR-005).
* **Gate thresholds** are ≥ rules; exact equality is **not** a tie.

## **4) Decisiveness labels (what appears in Result/Report)**

* **Decisive.** All gates **Pass**, and the national margin ≥ **VM-VAR-062** (pp), and **no** mediation/enclave/protected-override flags exist in the resulting mapping.
* **Marginal.** Gates **Pass**, but margin < **VM-VAR-062** **or** any **Mediation/Enclave/Protected-override** flags are present.
* **Invalid.** Any **gate fails** (quorum, majority, double-majority, symmetry) **or** input **Validation** fails.

The label and its **reason** are written verbatim into **Result** and shown in the Report.

## **5) Edge cases (explicit)**

* **Exact threshold hit** (e.g., support = 55.000% with threshold 55): **Pass**.
* **Include-blanks setting (VM-VAR-007):** Affects **gate denominators only**; tabulation/allocation remain on valid ballots/approvals/scores.
* **Zero votes in a Unit:** mark Unit **data\_ok=false**; it contributes no allocation; for gates, denominators follow §1.2 logic at aggregate level.
* **Missing eligible\_roll where quorum > 0:** **Validation** fails (run becomes **Invalid** with reasons).
* **Multiple frontier modes:** not permitted; exactly one or none.
* **Protected override on + symmetry:** symmetry applies to thresholds only; using an override does **not** change symmetry evaluation but **does** force a **Marginal** label due to flags.
* **Double-majority with frontier=none:** enforced rule to use by\_list/by\_tag (see §1.3).

## **6) Traceability**

* **Variables:** VM-VAR-020..029 (gates & symmetry), 040..048 (frontier & contiguity), 050..052 (ties), 060..062 (labels/marginal band).
* **Functions:** VM-FUN-006 (gates), VM-FUN-007 (frontier mapping), VM-FUN-008 (tie resolution), VM-FUN-009 (labeling).
* **DB:** reads **Unit.eligible\_roll**, **Adjacency.type**, **Option.order\_index**; writes flags into **Result.UnitBlock** and **FrontierMap**.

**Done:** Gate math (with eligible\_roll), double-majority constraints, symmetry (with exceptions), frontier mapping with contiguity/policies/protections, tie resolution order, label rules, and edge cases are all explicit and unambiguous.