

Master Thesis

Optimization of Inverter Placement and Cable Routing for Distributed Inverter Topologies in Solar Plants

Dumitru Verşebeniuc

Thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science of Artificial Intelligence
at the Department of Advanced Computing Sciences
of the Maastricht University

Thesis Committee:

Dr. Matúš Mihalák

Maastricht University
Faculty of Science and Engineering
Department of Advanced Computing Sciences

August 24, 2026

Abstract

Once the PV-panels—the PV-modules—of a solar plant are placed, its electrical design of a solar plant with the so-called distributed string-inverters topology couples six decisions: the partition of modules into PV-strings; the type and placement of the inverters; the assignment of strings to inverters; and the routing of DC-homerun and AC-feeder cables. Inverters, cables, and the trenches that carry them absorb a large share of a plant’s construction cost, yet industry practice fixes the six decisions sequentially by hand, and no prior optimisation model covers all six decisions within a single program on the irregular sites where such plants are built. This work formulates the joint problem as two monolithic mixed-integer linear programs over discretised routing graphs: a string-indexed model and an inverter-type-indexed model. The second is an approximation: it gives up the upper bound on each individual string-to-inverter circuit’s length, bounding instead its average over the strings of each inverter. It further contributes nine benchmark instances—operating plants of 1 to 10 MW, transcribed from engineering drawings and each annotated by hand with an engineer-style reference design—against which this and future layout algorithms can be evaluated. Solved directly, the more scalable of the two models produces a layout on only one of the nine plants within an hour. Fixing decision groups to the engineer’s reference values one group at a time reveals the bottleneck: the partition of modules into PV-strings. A four-stage decomposition built on that diagnosis—the topology fixed by a single-commodity module flow, module-to-inverter paths as a multi-commodity flow, the full model under the frozen topology, and an unfrozen polish—produces complete layouts on all nine plants without any annotation, each priced 0.3–5.2% below the reference design under the model’s objective. On the six smaller plants it reproduces the reference design’s exact inverter and string counts in layouts that at plant scale are hard to distinguish from the hand designs, while routing 10–34% less DC-homerun cable (18.3 km in aggregate) and 2–14% less PV-string cable—savings hidden in terminal orientations and string-to-inverter pairings the eye does not resolve. On the three larger plants it instead revises the design, removing inverters or trenches at the price of longer homeruns—trades that improve the model’s objective, though their worth under market prices remains open. Code and benchmark instances are available at <https://github.com/DikaVer/milp-solar-optimizer-public>.

Contents

1	Introduction	4
1.1	Motivation	4
1.2	Different Layouts of Solar Plants	5
1.2.1	Traditional Central-Inverter Architecture	5
1.2.2	Distributed Inverter Architecture	6
1.3	Inverter Placement and Cable Routing in PV Plants	7
1.3.1	DC Cable Routing on a Fixed String Partition	7
1.3.2	Sequential Frameworks	8
1.3.3	Integrated Inverter Placement and AC/DC Cable Routing Formulations	8
1.3.4	Research Gap and Positioning	8
2	Problem definition	10
2.1	Glossary	10
2.2	Modelling Assumptions	11
2.2.1	Single-Row Arrays	11
2.2.2	Module Interconnection Cables	11
2.2.3	Topological Cable Routing	11
2.3	Problem Instance	11
2.3.1	Geometry and Metrics	12
2.3.2	Edge Categories	12
2.3.3	DC Graph	12
2.3.4	Boundary Polygon	13
2.3.5	AC Graph	13
2.3.6	Edge Weights	13
2.4	Geometric Conflict Sets	14
2.5	Feasible Solution	15
2.5.1	Inverter Placements	15
2.5.2	PV-Strings	15
2.5.3	String-to-Inverter Assignments	16
2.5.4	DC-Homeruns	16
2.5.5	Inverter-to-Transformer Assignments	16
2.5.6	AC-Feeders	16
2.5.7	Geometric Non-Crossing Constraint	16
2.6	Cost of a Feasible Solution	17
2.6.1	PV-String Edges	17
2.6.2	DC-Homerun Edges	17
2.6.3	AC-Feeder Trenches	17
2.6.4	DC-Trench Cost	18
2.6.5	Inverter Cost	18
2.6.6	AC/DC Overlap Penalties	18
2.6.7	DC-Trench / AC-Internal Overlap Penalties	18

3	Problem modeling with PV-string indices	19
3.1	Graph Transformation	19
3.2	Model Parameters	19
3.3	Decision Variables and Constraints	20
3.3.1	Inverter Placement	20
3.3.2	PV-Strings	21
3.3.3	Topological Routing	22
3.3.4	String-to-Inverter Assignments	23
3.3.5	DC-Homeruns	23
3.3.6	DC Circuit Length	24
3.3.7	Inverter-to-Transformer Assignments	24
3.3.8	AC-Feeders	25
3.3.9	DC Trenching and Conflicts	26
3.3.10	AC/DC Overlaps	26
3.3.11	Objective Function	27
4	Problem modeling with inverter type indices	28
4.1	Decision Variables and Constraints	28
4.1.1	Inverter Placement	28
4.1.2	PV-Strings	29
4.1.3	Topological Routing	29
4.1.4	String-to-Inverter Assignments	30
4.1.5	DC-Homeruns	31
4.1.6	DC Circuit Length	31
4.1.7	Inverter-to-Transformer Assignments	32
4.1.8	AC-Feeders	33
4.1.9	DC Trenching and Conflicts	34
4.1.10	AC/DC Overlaps	35
4.1.11	Objective Function	35
5	Monolithic Solver Evaluation	36
5.1	Experimental Setup	36
5.2	Scaling on Synthetic Instances	37
5.2.1	Synthetic Instance Family	37
5.2.2	Protocol	38
5.2.3	Model Size	38
5.2.4	LP Relaxation	38
5.2.5	Scaling Results	40
5.3	Where the Monolithic MILP Struggle	41
5.3.1	Instances and Reference Annotations	41
5.3.2	Freeze Hierarchy and Protocol	41
5.3.3	Results	42
6	Staged Decomposition Solver	44
6.1	Stage 1: Aggregated Flow MILP	44
6.2	Stage 2: Path Resolution MILP	46
6.3	Stage 3: Topology-Frozen Chapter 4 MILP	48
6.4	Stage 4: Polished Chapter 4 MILP	49
6.5	Decomposition Summary	49
6.6	Evaluation Against the Human Annotation	49
6.6.1	Protocol	50
6.6.2	Total Cost	50
6.6.3	Cable Lengths and Inverter Counts	51
7	Conclusion	53
7.1	Summary of Findings	53
7.2	Limitations	54
7.3	Statement on the Use of AI Tools	55

A	Choice of cost coefficients	58
B	Synthetic instance geometry	59
C	Scaling study detail	60
C.1	Constraint growth by family	60
C.2	Per-cell results	61
D	Freezing study detail	63
D.1	Model size on the real instances	63
D.2	Per-cell solver outcomes	63
E	Real plant layouts	65
F	Staged solver detail	75
F.1	Objective cost breakdown	75
F.2	Cable lengths and inverter counts	76
F.3	Per-stage solver detail	76
F.4	Layouts of all nine instances	77

Chapter 1

Introduction

1.1 Motivation

In 2025, 511 gigawatts (GW) of new solar photovoltaic (PV) capacity was installed worldwide—about three quarters of all renewable power added that year [1]—and PV is expected to remain the largest single source of new renewable capacity through the decade [2]. The expansion is cost-driven: in 2024 the global cost of electricity from new PV plants fell to roughly USD 0.043 per kilowatt-hour, some 41% below the cheapest fossil-fuel alternative [3]. These cost declines, however, are not spread evenly across a plant’s components.

PV panels no longer dominate plant cost. Between 2010 and 2024, the total installed cost of a utility-scale PV plant fell by roughly 87%, from about USD 5,300 to USD 690 per kilowatt [3], almost entirely through the collapse in module prices. The remaining components of the PV plant: inverters, transformers, cables, and trenches that carry the modules’ output to the grid, together with the civil and electrical works to install them—now account for the bulk of a new plant’s cost [4, 5]. In a study with the State Grid Corporation of China (SGCC), Wang et al. [6] report that material handling—the cables, trenches, and on-site civil works for the electrical network—exceeds 60% of total construction cost, with cable routing the dominant component.

This cost structure is more pronounced at smaller scale. Europe’s 65.1 GW of new PV in 2025 [7] came largely from *small-to-medium-scale plants* of 1–10 megawatts (MW) peak power, not the multi-hundred-megawatt plants that dominate China, India, and the United States [5]. Large flat sites are scarce across Europe [7]. New plants are therefore built on whatever land is available—hillsides, old farming terraces, brownfields, and other small or irregular plots—where the rectangular block layouts of large plants no longer fit. European sites push costs further: the panels must be placed in shorter, broken-up rows, more inverters are needed, and the cables must detour around obstacles. Inverters, cables, and trenches therefore take a larger share of the cost.

Despite this economic weight, the electrical layout of a plant is still drawn by hand. Engineers work in a fixed order. Once the panels—the PV-panels—of a solar plant are placed, engineers choose where to put the inverters—the devices that convert the direct current (DC) produced by the panels into the alternating current (AC)—guided by rules of thumb and simple geometric arguments. They then group the panels into *strings*, chains of panels wired one after another so that each chain feeds a single inverter. They finally draw the cable routes by hand in a computer-aided design (CAD) environment [6].

This locked-in pipeline has a measurable cost. A sequential decomposition cannot capture the interaction between string formation, inverter placement, and cable routing: inverter positions that minimise DC-cable length may force long AC-cable length, and a routing that simplifies trenching may demand suboptimal string-to-inverter assignments. Wang et al. [6] quantify this gap on 50 real-world Chinese plants of 3–30 MW, where integrated optimisation cuts total cost by an average of 19.3% relative to SGCC’s manual method.

However, Wang et al.’s cost saving was obtained for a plant that build on large flat sites, where a few large inverters stand at points chosen in advance and only the cabling is optimised, this type of plants called the *central-inverter architecture*. The plants now being built across Europe are different: many small inverters are spread over the field, each one serving a small group of panels, this type of plants called the *distributed-inverter architecture*. Distributed inverter architecture leaves far more open than the cabling alone. The engineer must decide how many inverters to install, of which type, where to place them, which panels each of them serves, and where the cables run. None of these decisions can be taken on its

own, because each one changes the cost of the rest—and for plants of distributed-inverter architecture, no published method, where decisions are made together.

The objective of this work is to close that gap: given a site, its panels, and the obstacles on it, compute the electrical layout of lowest total cost—the number and positions of the inverters, the grouping of the panels into strings, and the cable routes—as the solution of one optimisation problem rather than a separate decisions. We write these decisions as a single mixed-integer linear programme (MILP) whose objective is the combined cost of inverters, cables, and trenching, and we then split that programme into a sequence of smaller problems, carrying the cost of the decisions to optimality.

1.2 Different Layouts of Solar Plants

A PV panel, also called PV module, produces DC power, but the grid operates on AC, so every plant needs *inverters* between DC and AC; their number, size, and placement define its electrical architecture. Two architectures dominate practice: a *central-inverter* architecture, where a few large inverters collect the DC power of many strings at one point, and a *distributed-inverter* architecture, where each string assigned to its own small inverter near the modules. This section describes both architectures and explains why the distributed layout suits the plants studied here.

1.2.1 Traditional Central-Inverter Architecture

Long-time the standard for utility-scale plants, this architecture is assumed by most previous cable-routing studies in the *Solar Farm Cable Layout Problem* (SoFaCLaP) family [8]. A few high-power *central inverters* collect the DC power of many strings at a single point; en route, the DC cables pass through Y-connectors and combiner boxes that bundle them in stages. Figure 1.1 shows the six layers involved.

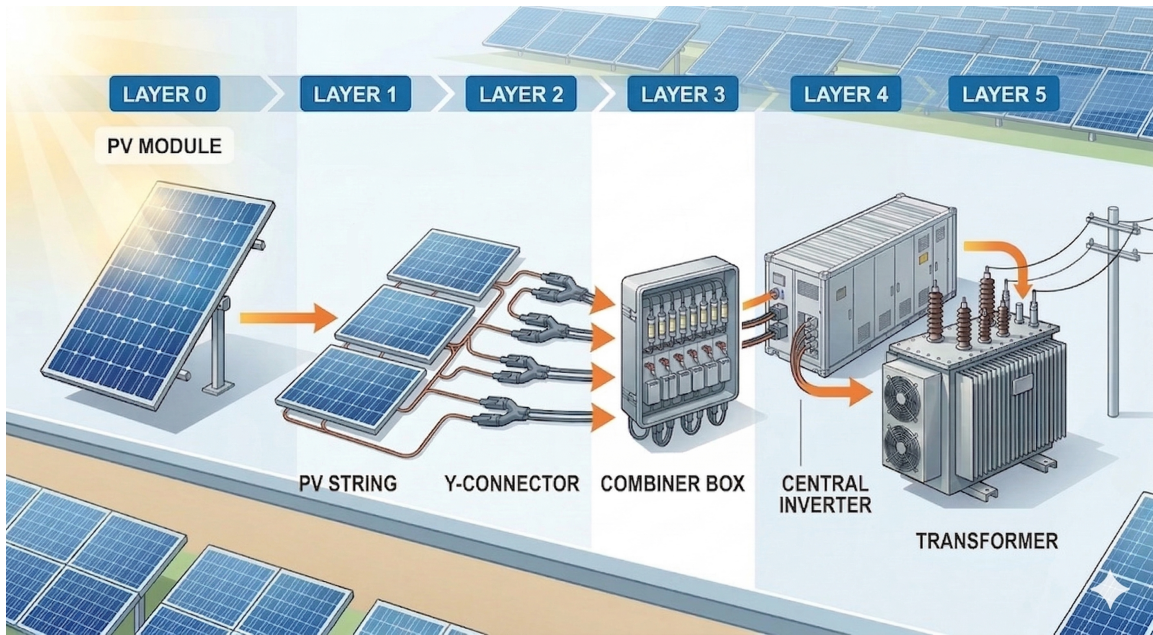


Figure 1.1: The six layers of the traditional central-inverter architecture. Power flows from left to right. Layer 0: a *PV module* generates DC current from sunlight. Layer 1: modules connected in series form a *PV string*. Layer 2: a *Y-connector* joins two strings into a single cable pair. Layer 3: a *combiner box* bundles the cables of many strings into one thick DC cable. Layer 4: a *central inverter* converts the combined DC power into three-phase AC. Layer 5: a *transformer* raises the AC voltage to the level used by the plant’s medium-voltage collection network.

A central inverter typically has a single maximum power point tracking (MPPT) channel, forcing all connected strings to the same DC voltage. If one string underperforms—through shade, dirt, or module tolerances—the whole group loses power with it.

1.2.2 Distributed Inverter Architecture

The distributed-inverter architecture, studied throughout this work, drops the Y-connectors and combiner boxes: each PV string is wired directly to its own small inverter placed close to the modules. The resulting hierarchy has only four levels (Figure 1.2).

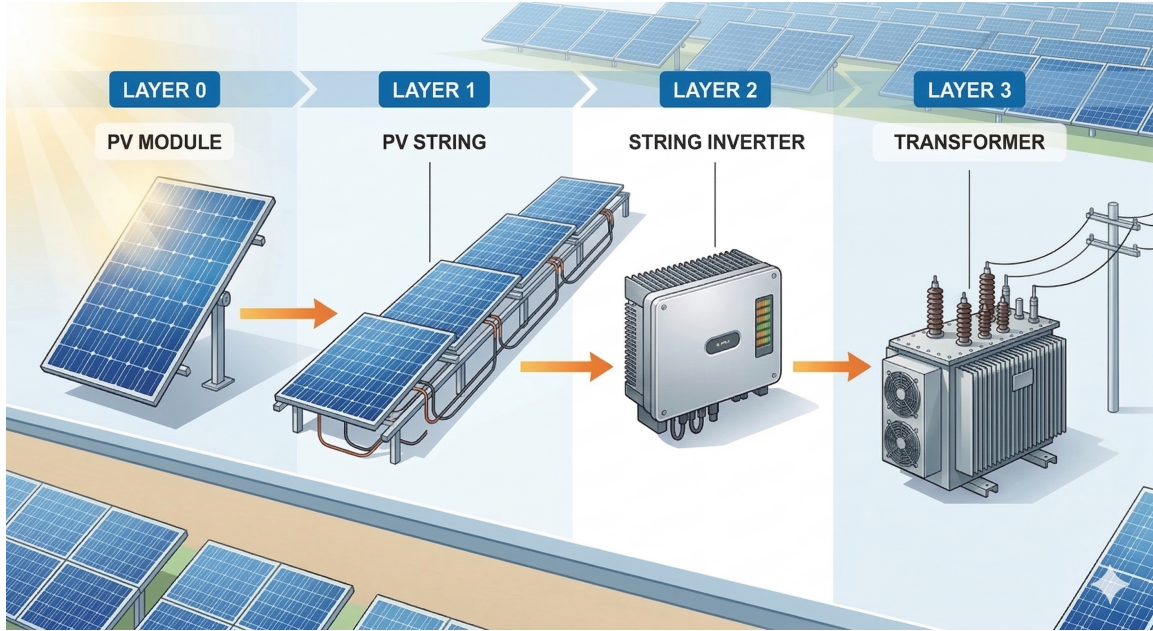


Figure 1.2: The four layers of the distributed-inverter architecture used in this work. Layer 0: a *PV module* generates DC current from sunlight. Layer 1: the modules of a single row are connected in series to form a *PV string*. Layer 2: the string is connected directly to a small, compact *string inverter* that converts its DC output into three-phase AC close to the modules. Layer 3: the AC output of many string inverters is collected and raised to medium voltage by the plant’s *transformer*.

Dropping the combiner-box layer removes more than the boxes: with them go the DC fuses and disconnect switches, the thick feeder cables, and the concrete foundations and monitoring each box required. In return, the plant needs many more inverters per megawatt, and every PV string keeps its own pair of DC cables—a positive and a negative homerun—running directly to its assigned inverter.

The choice between two architectures depends on the terrain. Figure 1.3 contrasts a flat-terrain central-inverter plant (left) with an irregular-terrain European hillside plant (right).



(a) Flat-terrain central-inverter plant.



(b) Irregular-terrain distributed-inverter plant.

Figure 1.3: Two representative sites at the scale of the two architectures. *Left:* a flat-terrain central-inverter park with PV modules in long parallel blocks of uniform orientation, separated by a regular rectangular corridor grid. *Right:* a small-to-medium-scale distributed-inverter plant on an irregular European hillside whose array rows follow the terrain and vary in length and orientation.

On a site like the right one of Figure 1.3, the central-inverter architecture is impractical for two reasons. First, its shared MPPT is efficient only when strings match in orientation and tilt—a condition that fails where rows vary in both, making mismatch losses substantial. Second, with modules scattered over

irregular terrain, the DC cable length to a single conversion point grow long enough to dominate the cost the architecture is meant to save. The distributed architecture avoids both by construction: each string keeps its own MPPT, so voltage matching is needed only within a string, and inverters can sit wherever a local cluster of modules exists.

The plants studied here are European sites of exactly this irregular kind, so the rest of the work adopts the four-level distributed architecture. Figure 1.4 shows a complete layout of distributed-inverter plants: PV-modules in spatially continuous rows—the *PV-arrays*—wired in series into *PV-strings*, each ending in a positive and a negative *terminal*. From these, two *DC-homerun* cables run to the assigned string inverter, which converts the DC to AC and feeds it through an *AC-feeder* to the plant’s *transformer*. Cables share excavated *DC-trenches* and must avoid *obstacles* while staying inside the site *boundary*.

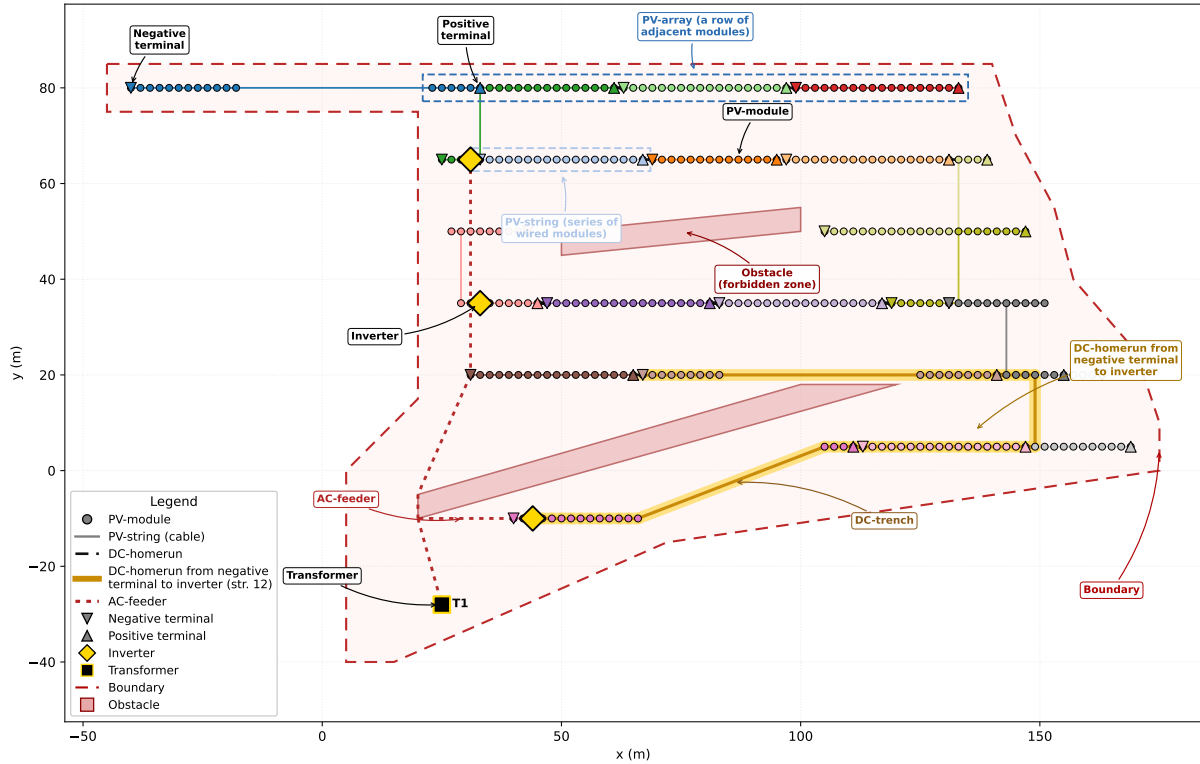


Figure 1.4: A complete layout of a small-to-medium-scale distributed-inverter plant of the kind targeted in this work. PV-modules are grouped into rows (*PV-arrays*) and wired in series into *PV-strings*; each string is connected by two *DC-homerun* cables—one from its positive and one from its negative terminal—to a nearby string *inverter*, whose AC output is carried by *AC-feeders* to the plant’s *transformer*. Cables share excavated *DC-trenches* and must avoid *obstacles* (forbidden zones) while remaining inside the site *boundary*.

1.3 Inverter Placement and Cable Routing in PV Plants

The joint optimization of string partition, inverter placement, string–inverter assignment, and DC/AC routing has followed three lines: fixing the string partition and optimizing only DC cables; decomposing the design into a sequence of inverter placement, and cable routing subproblems; and formulating all decisions in a single mathematical program. We review each below, then identify the gap for the distributed-inverter plants of Section 1.2.2.

1.3.1 DC Cable Routing on a Fixed String Partition

Fixing the string partition reduces plant design to routing alone. Gritzbach et al. [8] introduce the SoFaCLaP—choosing a cable type for every edge of a layered graph and activating intermediate vertices under vertex-capacity and flow-conservation constraints—and prove it strongly NP-complete. Bläsius et al. [9] isolate its combinatorial core as the *Constrained Layer Tree* problem, solved by a dynamic program orders of magnitude faster than a generic MILP solver. Löhken and Stiglmayr [10] revisit the cable-trench

problem, trading spanning-tree (trench) cost against root-to-vertex path (cable) cost; their bi-objective formulation finds compromise trees that no weighted-sum scalarisation can. All three refine routing on a fixed PV demand, but none decide how modules are grouped into strings.

1.3.2 Sequential Frameworks

Sequential frameworks produce a complete design by solving the subproblems in a fixed order. Cortez et al. [11] fix inverter placements, module-to-inverter assignment, and string partition by two-level k -means, then route the DC and AC networks in two successive MILPs; their 1.2 MW Brazilian case study respects voltage-drop and current limits, but selects inverter sites on a geometric criterion uncoupled from cable cost. Khatib and Assi [12] follow similar approach with a four-stage heuristic, reporting a 26.9% cable-length reduction on a 900 kWp Hebron plant. The ACM-PSO optimizer of Zhang et al. [13] optimises inverter placements and string-to-inverter assignment with respect to a routing cost computed from a pre-fixed trench skeleton. The review of Zidane et al. [5] confirms the dominance of metaheuristics and flags integrated multi-objective formulations as open. In every studied variant inverter placement is decided before cable routing, so cable routing cost cannot revise placement.

1.3.3 Integrated Inverter Placement and AC/DC Cable Routing Formulations

Integrated formulations decide inverter placement and AC/DC cable routing in a single program. The closest is the *integrated location and routing* (ILR) problem of Wang et al. [6], which presumes a central-inverter architecture on plants pre-partitioned into rectangular districts under a Manhattan metric rather than the Euclidean geometry of irregular sites. Wang et al.’s work jointly chooses internal access roads and inverter locations, combiner-box positions and types, however the string partition and inverter-to-transformer feeders fall outside its scope; a customised Benders decomposition terminates within 0.02% of optimality on average and cuts cost by roughly 20% over State Grid Corporation of China manual designs on 50 real-world instances. Yao et al. [14] embed inverter placement in a differential-evolution metaheuristic with a k -means/ A^* decoder, but fix inverter and combiner-box counts and likewise assume a central-inverter architecture. Neither work transfers to the distributed-inverter plants of Section 1.2.2 without substantial redesign.

1.3.4 Research Gap and Positioning

Table 1.1 summarises which design decisions each of the closest references includes as optimization targets and which algorithmic technique is used to solve the corresponding model.

Reference	String partition	Inverter placement	Inverter type/number	Inverter assignment	DC routing	AC routing	Method
Gritzbach et al. [8]		✓		✓	✓		MILP
Bläsius et al. [9]		✓		✓	✓		DP
Löhken and Stiglmayr [10]					✓		MILP (bi-obj.)
Cortez et al. [11]	✓	✓	✓	✓	✓	✓	MILP (3-stage)
Khatib and Assi [12]		✓	✓	✓	✓	✓	Heuristic
Yao et al. [14]		✓	✓	✓	✓	✓	DE + decoder
Zhang et al. [13]		✓		✓			PSO
Wang et al. [6]		✓		✓	✓		Benders + B&C
This work	✓	✓	✓	✓	✓	✓	MILP + decomp.

Table 1.1: Design decisions modeled as variables in the closest references and in this thesis. A checkmark indicates that the corresponding decision is part of the optimization model; an empty cell indicates that it is either fixed externally or omitted. “String partition” refers to deciding which modules belong to the same series-connected string; “Inverter assignment” to deciding which strings connect to which inverter. The “Method” column records the primary algorithmic technique used to solve the corresponding model.

Two observations follow. First, no prior work *both* covers the six decisions (string partition; inverter type, location, and assignment; DC-homerun and AC-feeder routing) *and* solves them within a single program with feedback between placement and routing: Cortez et al. [11] attain full scope, but across three sequential stages. Second, while Wang et al. [6], covers three of the six decisions as a joint problem,

it presumes a central-inverter architecture on plants pre-partitioned into rectangular districts, which is incompatible with the distributed-inverter plants on irregular Euclidean sites considered here. This work targets both gaps: full decision scope within a single program, and applicability to distributed-inverter plants on irregular-terrain real-world sites.

The question is therefore whether the six decisions can be made jointly, by one optimisation model, on the distributed-inverter plants. Three artefacts answer it. Two monolithic MILP formulations (Chapters 3 and 4) construct the joint problem of Chapter 2 as a single mixed-integer linear program. A benchmark of nine real-world plants, transcribed from engineering drawings and each annotated by hand with an engineer-style reference design; the layouts, their annotations, and the solver code with experiments are released open-source (Chapter 5). A four-stage decomposition (Chapter 6), where stages run in sequence, each fixing part of the solution for the next, but the joint scope is recovered at the end: the final stage releases those fixings and re-optimises every decision in monolithic MILP from the incumbent it inherits. The four-stage decomposition solutions are measured against the human annotated solutions.

Chapter 2

Problem definition

Section 2.1 establishes the terminology used throughout the thesis, and Section 2.2 states the modelling assumption. Section 2.3 turns the plant’s physical components—PV-modules, arrays, and transformers—together with the inverter types and global capacity parameters into a DC routing graph, an AC routing graph, and a boundary polygon for the plant perimeter. Section 2.4 prunes boundary-violating edges and pre-computes the geometric conflict sets: DC–DC crossings, treated as a hard planarity rule, and AC/DC and DC-trench/AC-internal overlaps, treated as soft penalties. Section 2.5 defines a feasible solution as a tuple of these decisions—inverter placement, the partition of modules into strings, the string-to-inverter and inverter-to-transformer assignments, and the cable routes—and lists the validity rules each must satisfy. Section 2.6 defines the cost of a feasible solution as the sum of cable, trenching, and inverter costs plus the overlap penalties.

2.1 Glossary

The model rests on a small vocabulary of physical objects—modules, arrays, strings, terminals, inverters, homeruns, feeders, and trenches—together with one geometric event, a *conflict*. Each is defined below; the formal symbols introduced in the next section.

PV-module or module: The smallest photovoltaic unit at the site. In the graph abstraction, each module is a vertex at which cable connections can originate or terminate.

PV-array or array: A geometrically aligned sequence of PV-modules.

PV-string or string: A series connection of PV-modules whose voltages add and whose combined output feeds a single inverter.

Inverter: A device that converts the DC output of PV-strings into AC power. Each inverter type imposes bounds on the number of strings it can accept and on the number of modules per string.

DC-homerun or homerun: A cable connecting a terminal of a PV-string to its assigned inverter. Each string requires two homeruns, one from each terminal.

Terminals: The two end modules of a PV-string. Each terminal is the connection point for one of the two DC-homerun cables (positive or negative).

AC-feeder or feeder: A cable connecting an inverter to its assigned transformer.

Trench: A physical excavation hosting one or more parallel cables along an edge. Per-trench capacity bounds limit how many cables fit in one trench; additional parallel trenches are dug when more cables are routed through it.

Trail: A walk in a graph that repeats no edge: an ordered sequence of vertices in which every consecutive pair is an edge, and no edge is used twice. Vertices may repeat. In a directed graph the condition is that no *arc* repeats, so an undirected edge may be traversed at most twice, once in each orientation. The cable of a PV-string is a directed trail (Section 2.5); the cables of a homerun and of an AC-feeder are paths, trails that repeat no vertex either.

Geometric conflict: A geometric event in which two cables intersect away from a shared endpoint, or a cable crosses the plant boundary.

2.2 Modelling Assumptions

2.2.1 Single-Row Arrays

Each PV-array carries two parallel rows of PV-modules; we model one row and replicate its string structure verbatim on the second. This is a simplification, not a property of the problem: it rules out designs in which the two rows belong to different strings or feed different inverters.

2.2.2 Module Interconnection Cables

Adjacent modules in a string are joined by their own cables, already bundled into the module price. We nonetheless charge each such connection at the PV-string cable cost rather than crediting cables as free, because a zero-cost connection would be selected wherever available and would then have to be pruned from the final solution. Charging cables avoids that post-processing, at the price of a slight over-estimate of cabling cost. This too is a simplification, not a property of the problem.

2.2.3 Topological Cable Routing

In practice cables follow continuous trenched paths constrained by access roads, module rows, and the boundary. We approximate each path by a topological discretization: a path in the DC or AC graph whose length is the sum of Euclidean distances between consecutive vertices. The DC graph mirrors the physical sequence of modules in a string, so the discretization closely matches reality; the AC graph is more abstract. Computed routing in the AC graph is a coarser approximation of reality than routing in the DC graph, and in practice the real-world path for AC-feeders will be shorter than the route produced by the discretized AC graph. The discretization therefore overestimates AC-feeder cable length, so the resulting cost figures are conservative rather than exact.

2.3 Problem Instance

We model the site as a geometric graph. The following table lists the components and parameters fixing a problem instance.

Symbol	Meaning
$\mathcal{M} = \{1, \dots, N_{\mathcal{M}}\}$	Set of discrete <i>PV-modules</i> available at the site.
$\mathcal{A} = \{A_1, \dots, A_{N_{\mathcal{A}}}\}$	The set of $N_{\mathcal{A}}$ PV-arrays. The arrays partition $\mathcal{M}$, i.e., $\bigcup_i A_i = \mathcal{M}$ and $A_i \cap A_j = \emptyset$ for $i \neq j$, and each array is an ordered sequence of spatially adjacent modules: $A_i = \langle m_{i,1}, m_{i,2}, \dots, m_{i, A_i } \rangle$.
$\mathcal{T} = \{1, \dots, N_{\mathcal{T}}\}$	Set of available <i>transformer</i> locations.
$\mathcal{K} = \{1, \dots, N_{\mathcal{K}}\}$	The set of available inverter types. Each type $k \in \mathcal{K}$ is characterised by string-capacity bounds $[S_k^{\min}, S_k^{\max}]$ (number of strings) and string-length bounds $[Q_k^{\min}, Q_k^{\max}]$ (modules per string).
N_{str}	Upper bound on the number of PV-strings, attained when every string has the minimum allowable length across all inverter types: $N_{\text{str}} = \left\lfloor \frac{N_{\mathcal{M}}}{\min_{k \in \mathcal{K}} Q_k^{\min}} \right\rfloor.$
$N_{\mathcal{K}}^{\max}$	Maximum number of distinct inverter types permitted in a single solution.
J_t	Capacity of transformer $t \in \mathcal{T}$: the maximum number of PV-modules it can serve.
$L_{\text{DC}}^{\max}$	Upper bound on the total DC-circuit length (PV-string cable plus both DC-homerun cables), imposed to limit voltage drop.
$N_{\text{DC}}^{\text{trench}}$	Maximum number of DC cables (PV-string or DC-homerun) that may share a single cross-array trench.
$N_{\text{AC}}^{\text{trench}}$	Maximum number of AC-feeder cables that may share a single AC trench.

2.3.1 Geometry and Metrics

The spatial configuration of the plant is defined by Euclidean coordinates in $\mathbb{R}^3$ and the associated length metric.

Symbol	Meaning
$\text{pos}(u)$	The Euclidean coordinate (x, y, z) of any component $u \in \mathcal{M} \cup \mathcal{T} \cup V^{\text{BND}}$, where V^{BND} denotes the boundary-polygon vertices defined below.
$\text{seg}_{\text{open}}(u, v)$	Open straight-line segment from $\text{pos}(u)$ to $\text{pos}(v)$, excluding the endpoints: $\text{seg}_{\text{open}}(u, v) = \{ \text{pos}(u) + \lambda(\text{pos}(v) - \text{pos}(u)) \mid \lambda \in (0, 1) \}$.
$\text{seg}_{\text{closed}}(u, v)$	Closed straight-line segment from $\text{pos}(u)$ to $\text{pos}(v)$, including the endpoints: $\text{seg}_{\text{closed}}(u, v) = \{ \text{pos}(u) + \lambda(\text{pos}(v) - \text{pos}(u)) \mid \lambda \in [0, 1] \}$.
$w(P)$	Total length of a vertex sequence $P = \langle v_1, \dots, v_k \rangle$, measured as the sum of Euclidean distances between consecutive vertices: $w(P) = \sum_{i=1}^{k-1} \ \text{pos}(v_i) - \text{pos}(v_{i+1})\ _2$.

2.3.2 Edge Categories

Every potential DC edge $e = \{u, v\}$ falls into one of two categories, determined by the array membership of its endpoints:

Symbol	Meaning
E^{int}	Internal edges: connections between adjacent modules within the same array. $E^{\text{int}} = \{ \{m_{i,j}, m_{i,j+1}\} \mid i = 1, \dots, N_{\mathcal{A}}; j = 1, \dots, A_i - 1 \}$.
E^{cross}	Cross-array edges: connections between modules in disjoint arrays. $E^{\text{cross}} = \{ \{m_{i,j}, m_{k,l}\} \mid i, k = 1, \dots, N_{\mathcal{A}}; i \neq k; j = 1, \dots, A_i ; l = 1, \dots, A_k \}$.

For an edge $e = \{u, v\}$ we write $\text{seg}_{\text{open}}(e) := \text{seg}_{\text{open}}(u, v)$ and $\text{seg}_{\text{closed}}(e) := \text{seg}_{\text{closed}}(u, v)$.

2.3.3 DC Graph

DC Graph Pruning: The complete cross-array edge set E^{cross} contains many candidates that are physically unrealisable: edges too long to be cabled, edges that intersect other arrays, or redundant edges between the same pair of arrays. We retain a subset $E_{\text{prune}}^{\text{cross}}$ via a three-step geometric heuristic, parameterised by a configured search radius $L_{\text{radius}}^{\text{DC}} > 0$ that caps the Euclidean length of any retained edge.

(i) *Target identification.* For each array A_i , define the *array baseline* $\mathcal{L}_i = \text{seg}_{\text{closed}}(m_{i,1}, m_{i,|A_i|})$. For a module $u \in A_i$ and another array A_k , write $\mathcal{N}_k(u, L_{\text{radius}}^{\text{DC}}) = \{ v \in A_k \mid \|\text{pos}(u) - \text{pos}(v)\| \leq L_{\text{radius}}^{\text{DC}} \}$ for the modules of A_k within the search radius of u . The target set retains, for each other array, the single nearest candidate:

$$\text{Target}(u) = \left\{ \arg \min_{v \in \mathcal{N}_k(u, L_{\text{radius}}^{\text{DC}})} \|\text{pos}(u) - \text{pos}(v)\| \mid k \in \{1, \dots, N_{\mathcal{A}}\} \setminus \{i\} \right\}.$$

(ii) *Geometric filtering.* To prevent cables from crossing intervening arrays, an edge $e = \{u, v\}$ with $u \in A_i$, $v \in A_j$ is retained by the geometric filter when

$$v \in \text{Target}(u) \quad \text{and} \quad \text{seg}_{\text{open}}(e) \cap \mathcal{L}_k = \emptyset \quad \forall k \in \{1, \dots, N_{\mathcal{A}}\} \setminus \{i, j\}.$$

(iii) *Per-array deduplication.* For each target module v and each source array A_i with $v \notin A_i$, let $F(v, A_i) = \{u \in A_i : \{u, v\} \text{ passes step (ii)}\}$ denote the surviving candidates. When $|F(v, A_i)| > 1$, only the shortest edge is kept: $E_{\text{prune}}^{\text{cross}}$ retains $\{u^*, v\}$ with

$$u^* = \arg \min_{u \in F(v, A_i)} \|\text{pos}(u) - \text{pos}(v)\|_2.$$

A module v may still receive connections from multiple distinct source arrays.

The DC graph is $G^{\text{DC}} = (V^{\text{DC}}, E^{\text{DC}})$ with vertex set $V^{\text{DC}} = \mathcal{M}$ and edge set $E^{\text{DC}} = E^{\text{int}} \cup E_{\text{prune}}^{\text{cross}}$.

2.3.4 Boundary Polygon

The plant boundary and any interior obstacle zones are represented as closed polygons $B^{\text{BND}} = (V^{\text{BND}}, E^{\text{BND}})$. The vertex set V^{BND} collects the corners of the plant-boundary polygon and of every interior obstacle polygon; the edge set E^{BND} connects consecutive vertices along each polygon, forming one closed loop per polygon.

2.3.5 AC Graph

The AC graph routes AC-feeder cables from inverter locations to transformers, using PV-module and boundary-polygon vertices as intermediate waypoints. Its vertex set is $V^{\text{AC}} = \mathcal{M} \cup \mathcal{T} \cup V^{\text{BND}}$, and its edge set decomposes as

$$E^{\text{AC}} = E^{\text{int}} \cup E_{\text{AC}}^{\text{cross}} \cup E^{\text{BND}} \cup E_{\text{link}}^{\text{BND}} \cup E_{\text{link}}^{\text{trans}}.$$

The internal edges E^{int} are inherited from the DC graph. The cross-array edges $E_{\text{AC}}^{\text{cross}}$ are generated by the same three-step pruning of Section 2.3.3, with the AC search radius $L_{\text{radius}}^{\text{AC}}$ in place of $L_{\text{radius}}^{\text{DC}}$. The boundary perimeter edges E^{BND} are included without filtering. The remaining two sets are defined below.

Boundary Link Edges $E_{\text{link}}^{\text{BND}}$. Each boundary vertex $c \in V^{\text{BND}}$ is connected to the nearest module of each array within the AC search radius $L_{\text{radius}}^{\text{AC}}$, subject to the array-baseline filter. For each $c \in V^{\text{BND}}$ and each $A_i \in \mathcal{A}$ with $\mathcal{N}_i(c, L_{\text{radius}}^{\text{AC}}) \neq \emptyset$, let

$$v^* = \arg \min_{v \in \mathcal{N}_i(c, L_{\text{radius}}^{\text{AC}})} \|\text{pos}(c) - \text{pos}(v)\|.$$

If $\text{seg}_{\text{open}}(c, v^*) \cap \mathcal{L}_k = \emptyset$ for all $k \in \{1, \dots, N_{\mathcal{A}}\} \setminus \{i\}$, then $\{c, v^*\}$ is added to $E_{\text{link}}^{\text{BND}}$.

Transformer Link Edges $E_{\text{link}}^{\text{trans}}$. Each transformer $t \in \mathcal{T}$ is connected to (a) the nearest module of each array within $L_{\text{radius}}^{\text{AC}}$, and (b) every boundary vertex within $L_{\text{radius}}^{\text{AC}}$. Transformer-to-module edges exclude the target array from the baseline filter; transformer-to-boundary edges check all baselines. For each $t \in \mathcal{T}$ and each $A_i \in \mathcal{A}$ with $\mathcal{N}_i(t, L_{\text{radius}}^{\text{AC}}) \neq \emptyset$, let

$$v^* = \arg \min_{v \in \mathcal{N}_i(t, L_{\text{radius}}^{\text{AC}})} \|\text{pos}(t) - \text{pos}(v)\|.$$

If $\text{seg}_{\text{open}}(t, v^*) \cap \mathcal{L}_k = \emptyset$ for all $k \in \{1, \dots, N_{\mathcal{A}}\} \setminus \{i\}$, then $\{t, v^*\}$ is added to $E_{\text{link}}^{\text{trans}}$.

For each $t \in \mathcal{T}$ and each $c \in V^{\text{BND}}$ with $\|\text{pos}(t) - \text{pos}(c)\| \leq L_{\text{radius}}^{\text{AC}}$, the edge $\{t, c\}$ is added to $E_{\text{link}}^{\text{trans}}$ whenever $\text{seg}_{\text{open}}(t, c) \cap \mathcal{L}_k = \emptyset$ for all $k \in \{1, \dots, N_{\mathcal{A}}\}$.

2.3.6 Edge Weights

Both graphs share the edge-weight function $w : E^{\text{DC}} \cup E^{\text{AC}} \rightarrow \mathbb{R}_{>0}$, which assigns each edge its Euclidean length:

$$w(\{v_i, v_j\}) = \|\text{pos}(v_i) - \text{pos}(v_j)\|_2.$$

The two uses of w agree on a path: $w(\langle v_1, \dots, v_k \rangle) = \sum_{i=1}^{k-1} w(\{v_i, v_{i+1}\})$.

2.4 Geometric Conflict Sets

The conflict sets collect edge pairs whose segments share interior points. One set is a hard constraint (DC-DC planarity); two are scaffolding for boundary-edge removal and are emptied immediately after the pruning step below; the remaining two feed the cost as soft penalties for AC/DC and DC-trench/AC-internal overlaps.

Set	Definition
$\mathcal{O}^{\text{cross} \times \text{cross}}$	DC-DC crossings: pairs of cross-array DC edges that intersect. $\mathcal{O}^{\text{cross} \times \text{cross}} = \{\{e_1, e_2\} \subseteq E_{\text{prune}}^{\text{cross}} \mid e_1 \neq e_2, \text{seg}_{\text{open}}(e_1) \cap \text{seg}_{\text{open}}(e_2) \neq \emptyset\}.$
$\mathcal{O}^{\text{cross} \times \text{AC}}$	DC-AC crossings: intersections between cross-array DC edges and AC edges. $\mathcal{O}^{\text{cross} \times \text{AC}} = \{(e^{\text{cross}}, e^{\text{AC}}) \in E_{\text{prune}}^{\text{cross}} \times E^{\text{AC}} \mid \text{seg}_{\text{open}}(e^{\text{cross}}) \cap \text{seg}_{\text{open}}(e^{\text{AC}}) \neq \emptyset\}.$
$\mathcal{O}^{\text{BND} \times \text{AC}}$	Boundary AC violations (scaffolding; emptied after pruning below): AC edges (excluding the boundary perimeter itself) that cross the boundary polygons B^{BND} . $\mathcal{O}^{\text{BND} \times \text{AC}} = \{(b, e^{\text{AC}}) \in E^{\text{BND}} \times (E^{\text{AC}} \setminus E^{\text{BND}}) \mid \text{seg}_{\text{closed}}(b) \cap \text{seg}_{\text{open}}(e^{\text{AC}}) \neq \emptyset\}.$
$\mathcal{O}^{\text{BND} \times \text{DC}}$	Boundary DC violations (scaffolding; emptied after pruning below): DC edges that cross the boundary polygons B^{BND} . $\mathcal{O}^{\text{BND} \times \text{DC}} = \{(b, e^{\text{DC}}) \in E^{\text{BND}} \times E^{\text{DC}} \mid \text{seg}_{\text{closed}}(b) \cap \text{seg}_{\text{open}}(e^{\text{DC}}) \neq \emptyset\}.$
$\mathcal{O}^{\text{cross} \times \text{int}}$	DC-trench / AC-internal overlap: set flags physical overlaps between DC trenches (carried on cross-array edges) and AC cables (carried on internal edges) that meet at a common module. (defined in full below).

The first set uses unordered pairs because both elements are DC edges; the other four use ordered tuples to mark the role of each component.

Boundary Edge Filtering The two boundary sets $\mathcal{O}^{\text{BND} \times \text{DC}}$ and $\mathcal{O}^{\text{BND} \times \text{AC}}$ are scaffolding: they identify edges to remove from the graphs, after which the sets themselves are empty. The pruning step is

$$E^{\text{DC}} \leftarrow E^{\text{DC}} \setminus \{e^{\text{DC}} \in E^{\text{DC}} \mid \exists (b, e^{\text{DC}}) \in \mathcal{O}^{\text{BND} \times \text{DC}}\},$$

$$E^{\text{AC}} \leftarrow E^{\text{AC}} \setminus \{e^{\text{AC}} \in E^{\text{AC}} \setminus E^{\text{BND}} \mid \exists (b, e^{\text{AC}}) \in \mathcal{O}^{\text{BND} \times \text{AC}}\}.$$

After this filtering, G^{DC} and G^{AC} are boundary-safe by construction; only $\mathcal{O}^{\text{cross} \times \text{cross}}$ and $\mathcal{O}^{\text{cross} \times \text{AC}}$ remain as live conflict sets, with the trench/internal overlap set defined below.

DC-Trench / AC-Internal Overlap Set For each array A_i , the array baseline $\mathcal{L}_i$ partitions the ground plane (the xy -projection of $\mathbb{R}^3$) into two open half-planes H_i^+ and H_i^- . For each module $m \in A_i$, the cross-array edges incident to m split according to the half-plane containing their far endpoint:

$$E_{\text{prune}}^{\text{cross},+}(m) = \{\{m, v\} \in E_{\text{prune}}^{\text{cross}} \mid \text{pos}(v) \in H_i^+\},$$

$$E_{\text{prune}}^{\text{cross},-}(m) = \{\{m, v\} \in E_{\text{prune}}^{\text{cross}} \mid \text{pos}(v) \in H_i^-\}.$$

The internal edges incident to m are

$$E^{\text{int}}(m) = \{e \in E^{\text{int}} \mid m \in e\}.$$

Only interior modules—those with $|E^{\text{int}}(m)| = 2$ —can host an overlap. An overlap occurs at such a module when cross-array edges leave on both the H_i^+ and H_i^- sides (forming a DC trench that crosses through m) and both adjacent internal edges carry AC cable. The set collects every such configuration:

$$\mathcal{O}^{\text{cross} \times \text{int}} = \{(e^+, e^-, e_1^{\text{int}}, e_2^{\text{int}}) \mid \exists m \in \mathcal{M} : e^+ \in E_{\text{prune}}^{\text{cross},+}(m), e^- \in E_{\text{prune}}^{\text{cross},-}(m), \{e_1^{\text{int}}, e_2^{\text{int}}\} = E^{\text{int}}(m)\}.$$

A single module m may generate multiple 4-tuples when it has several edges on the H_i^+ and/or H_i^- sides.

2.5 Feasible Solution

A feasible solution is a tuple of design decisions:

$$\text{sol} = \left(\begin{array}{ll} \mathcal{I} \subseteq \mathcal{M} \times \mathcal{K}, & \text{(Inverter Placements)} \\ \{R_j\}_{j=1}^{N_{\text{str}}}, & \text{(PV-Strings)} \\ \{I(R_j)\}_{j=1}^{N_{\text{str}}}, & \text{(String-to-Inverter Assignments)} \\ \{R_j^+\}_{j=1}^{N_{\text{str}}}, & \text{(Positive DC-Homeruns)} \\ \{R_j^-\}_{j=1}^{N_{\text{str}}}, & \text{(Negative DC-Homeruns)} \\ \{T(I)\}_{I \in \mathcal{I}}, & \text{(Inverter-to-Transformer Assignments)} \\ \{R_i^{\text{AC}}\}_{i=1}^{|\mathcal{I}|}, & \text{(AC-Feeders)} \end{array} \right)$$

Each component, with its validity constraints, is defined below.

2.5.1 Inverter Placements

A subset $\mathcal{I} \subseteq \mathcal{M} \times \mathcal{K}$. Each placed inverter is a pair of a location $m \in \mathcal{M}$ and a type $k \in \mathcal{K}$; we write $I = (m, k) \in \mathcal{I}$ when a single symbol for the pair is convenient.

Constraint (Location Assignment): At most one inverter (of any type) is installed at any module location. Formally, for every $m \in \mathcal{M}$,

$$|\{k \in \mathcal{K} \mid (m, k) \in \mathcal{I}\}| \leq 1.$$

Constraint (Inverter Standardisation): Let $\mathcal{K}(\mathcal{I}) = \{k \in \mathcal{K} \mid \exists m \in \mathcal{M} : (m, k) \in \mathcal{I}\}$ denote the set of used inverter types. The number of distinct types must not exceed $N_{\mathcal{K}}^{\text{max}}$:

$$|\mathcal{K}(\mathcal{I})| \leq N_{\mathcal{K}}^{\text{max}}.$$

2.5.2 PV-Strings

A collection $\{R_j\}_{j=1}^{N_{\text{str}}}$, where N_{str} is an upper bound on the number of strings; fewer than N_{str} strings may be non-empty in a feasible solution.

Each non-empty string R_j is a directed trail in G^{DC} :

$$R_j = \langle m_{a_1^j, p_1^j}, m_{a_2^j, p_2^j}, \dots, m_{a_{q_j}^j, p_{q_j}^j} \rangle,$$

an ordered sequence of q_j modules in which every consecutive pair (m_t, m_{t+1}) is an edge of G^{DC} and no directed arc is repeated. Equivalently, each undirected edge $\{u, v\}$ may be traversed at most twice, once in each orientation. The first module is the *negative terminal* $m_j^- := m_{a_1^j, p_1^j}$, and the last is the *positive terminal* $m_j^+ := m_{a_{q_j}^j, p_{q_j}^j}$; the two must be distinct, $m_j^- \neq m_j^+$.

Vertex Classification: Write $V(R_j)$ for the set of modules appearing in the trail. $V(R_j)$ decomposes into two disjoint subsets: *owned* vertices $V^{\text{own}}(R_j)$ (modules that electrically belong to the string and contribute power) and *extender* vertices $V^{\text{ext}}(R_j)$ (modules used only to route the trail to its terminals when no direct path is available):

$$V^{\text{own}}(R_j) \cap V^{\text{ext}}(R_j) = \emptyset, \quad V(R_j) = V^{\text{own}}(R_j) \cup V^{\text{ext}}(R_j).$$

The owned vertices across all non-empty strings form a partition of the module set $\mathcal{M}$:

$$\bigcup_{j: R_j \neq \emptyset} V^{\text{own}}(R_j) = \mathcal{M}, \quad V^{\text{own}}(R_j) \cap V^{\text{own}}(R_{j'}) = \emptyset \quad \forall j \neq j'.$$

Empty string slots ($R_j = \emptyset$) have no vertices, no terminals, and no inverter assignment.

Because terminals carry the string's power to the homerun cables, they cannot be extender vertices:

$$m_j^-, m_j^+ \in V^{\text{own}}(R_j).$$

2.5.3 String-to-Inverter Assignments

An assignment map where each non-empty PV-string R_j is assigned to exactly one inverter $I(R_j) \in \mathcal{I}$. For an inverter I , let $\mathcal{R}_I = \{j \in \{1, \dots, N_{\text{str}}\} \mid R_j \neq \emptyset, I(R_j) = I\}$ denote the indices of strings assigned to I . Empty string slots remain unassigned.

Constraint (String Length): For each inverter $I = (m, k) \in \mathcal{I}$ and each assigned string R_j with $j \in \mathcal{R}_I$, the number of owned modules lies within the string-length bounds of type k :

$$Q_k^{\min} \leq |V^{\text{own}}(R_j)| \leq Q_k^{\max}.$$

Constraint (Inverter String Capacity): The number of strings assigned to each inverter $I = (m, k) \in \mathcal{I}$ lies within its string-capacity bounds $[S_k^{\min}, S_k^{\max}]$:

$$S_k^{\min} \leq |\mathcal{R}_I| \leq S_k^{\max}.$$

2.5.4 DC-Homeruns

For each non-empty PV-string R_j , two paths in G^{DC} connect its terminals to the assigned inverter $I(R_j)$:

- R_j^+ : positive homerun, from m_j^+ to $I(R_j)$.
- R_j^- : negative homerun, from m_j^- to $I(R_j)$.

Constraint (DC Circuit Length): The DC circuit length (string trail plus both homerun paths) must not exceed $L_{\text{DC}}^{\max}$:

$$w(R_j^+) + w(R_j^-) + w(R_j) \leq L_{\text{DC}}^{\max}, \quad \forall j \text{ with } R_j \neq \emptyset.$$

2.5.5 Inverter-to-Transformer Assignments

A mapping where each placed inverter $I \in \mathcal{I}$ is assigned to exactly one transformer $T(I) \in \mathcal{T}$.

Constraint (Transformer Capacity): For a transformer $t \in \mathcal{T}$, let $\mathcal{I}_t = \{I \in \mathcal{I} \mid T(I) = t\}$ denote the inverters assigned to t . The total number of PV-modules whose power reaches t must not exceed its capacity J_t :

$$\sum_{I \in \mathcal{I}_t} \sum_{j \in \mathcal{R}_I} |V^{\text{own}}(R_j)| \leq J_t, \quad \forall t \in \mathcal{T}.$$

2.5.6 AC-Feeders

A family of paths $\{R_i^{\text{AC}}\}_{i=1}^{|\mathcal{I}|}$, one per placed inverter. Enumerating $\mathcal{I}$ as $\{I_1, \dots, I_{|\mathcal{I}|}\}$, each R_i^{AC} is a path in G^{AC} from I_i to its assigned transformer $T(I_i)$.

2.5.7 Geometric Non-Crossing Constraint

Boundary crossings were already removed in Section 2.4, and AC/DC crossings are handled as cost penalties (Section 2.6). The remaining hard geometric rule is DC planarity: for any conflicting pair $\{e_1, e_2\} \in \mathcal{O}^{\text{cross} \times \text{cross}}$, at most one of the two edges may appear in the union of all string trails and homerun paths:

$$\left| \left(\bigcup_{j=1}^{N_{\text{str}}} E(R_j) \cup E(R_j^+) \cup E(R_j^-) \right) \cap \{e_1, e_2\} \right| \leq 1, \quad \forall \{e_1, e_2\} \in \mathcal{O}^{\text{cross} \times \text{cross}}.$$

2.6 Cost of a Feasible Solution

Write $E(R)$ for the edge set of a route R and $\vec{E}(R)$ for its directed edge multi-set (a directed edge appears once for each traversal, so an edge crossed twice in opposite directions contributes twice).

The total cost of a feasible solution is

$$\begin{aligned}
 C(\text{sol}) = & \underbrace{\sum_{j=1}^{N_{\text{str}}} \sum_{e \in \vec{E}(R_j)} c_e^{\text{pv}} w_e}_{\text{PV-string edges}} + \underbrace{\sum_{j=1}^{N_{\text{str}}} \sum_{e \in E(R_j^+) \cup E(R_j^-)} c_e^{\text{home}} w_e}_{\text{DC-homerun edges}} \\
 & + \underbrace{\sum_{e \in E_{\text{used}}^{\text{AC}}} c_e^{\text{AC}} \tau_e^{\text{AC}} w_e}_{\text{AC-feeder trenches}} + \underbrace{\sum_{e \in E_{\text{used}}^{\text{cross}}} c_e^{\text{trench}} \tau_e^{\text{DC}} w_e}_{\text{DC-trench cost}} \\
 & + \underbrace{\sum_{(m,k) \in \mathcal{I}} c_k^{\text{inv}}}_{\text{Inverter cost}} + \underbrace{\sum_{(e_{\text{DC}}, e_{\text{AC}}) \in \mathcal{O}_{\text{used}}^{\text{ov}}} c_{(e_{\text{DC}}, e_{\text{AC}})}^{\text{ov}}}_{\text{AC/DC overlap penalties}} + \underbrace{\sum_{\omega \in \mathcal{O}_{\text{used}}^{\text{tov}}} c_{\omega}^{\text{tov}}}_{\text{Trench/AC overlap penalties}}
 \end{aligned}$$

The subsections below define the cost coefficients ($c^{\text{pv}}, c^{\text{home}}, c^{\text{AC}}, c^{\text{trench}}, c^{\text{inv}}, c^{\text{ov}}, c^{\text{tov}}$), the trench counts ($\tau^{\text{AC}}, \tau^{\text{DC}}$), and the used-edge sets ($E_{\text{used}}^{\text{AC}}, E_{\text{used}}^{\text{cross}}, \mathcal{O}_{\text{used}}^{\text{ov}}, \mathcal{O}_{\text{used}}^{\text{tov}}$).

2.6.1 PV-String Edges

String trails are summed over the directed edge multi-set $\vec{E}(R_j)$ because a trail may traverse the same undirected edge twice in opposite directions, and each traversal lays a cable. Let $c_e^{\text{pv}} \geq 0$ be the unit PV-string cost on edge e . The total PV-string cabling cost is

$$C_{\text{pv}} = \sum_{j=1}^{N_{\text{str}}} \sum_{e \in \vec{E}(R_j)} c_e^{\text{pv}} w_e.$$

2.6.2 DC-Homerun Edges

The positive and negative homeruns R_j^+ and R_j^- are edge-disjoint by construction, so each edge in $E(R_j^+) \cup E(R_j^-)$ carries exactly one homerun cable. Let $c_e^{\text{home}} > 0$ be the unit homerun cost on edge e . The total DC-homerun cabling cost is

$$C_{\text{home}} = \sum_{j=1}^{N_{\text{str}}} \sum_{e \in E(R_j^+) \cup E(R_j^-)} c_e^{\text{home}} w_e.$$

2.6.3 AC-Feeder Trenches

Multiple AC-feeder cables can share a single trench, with each trench accommodating up to $N_{\text{AC}}^{\text{trench}}$ cables. Let

$$E_{\text{used}}^{\text{AC}} = \bigcup_{i=1}^{|\mathcal{I}|} E(R_i^{\text{AC}}) \subseteq E^{\text{AC}}$$

be the set of AC edges used by at least one feeder. For $e \in E_{\text{used}}^{\text{AC}}$, the number of feeders sharing e is

$$n_e^{\text{AC}} = \sum_{i=1}^{|\mathcal{I}|} \mathbf{1}[e \in E(R_i^{\text{AC}})],$$

and the number of AC trenches required on e is the ceiling

$$\tau_e^{\text{AC}} = \left\lceil \frac{n_e^{\text{AC}}}{N_{\text{AC}}^{\text{trench}}} \right\rceil.$$

With unit trench cost $c_e^{\text{AC}} > 0$, the total AC-feeder trenching cost is

$$C_{\text{AC}} = \sum_{e \in E_{\text{used}}^{\text{AC}}} c_e^{\text{AC}} \tau_e^{\text{AC}} w_e.$$

2.6.4 DC-Trench Cost

Multiple DC cables (PV-string cables and DC-homerun cables) can share a single cross-array trench, with each trench accommodating up to $N_{\text{DC}}^{\text{trench}}$ cables. Let

$$E_{\text{used}}^{\text{cross}} = E_{\text{prune}}^{\text{cross}} \cap \left(\bigcup_{j=1}^{N_{\text{str}}} (E(R_j) \cup E(R_j^+) \cup E(R_j^-)) \right)$$

be the cross-array edges carrying at least one DC cable. For $e \in E_{\text{used}}^{\text{cross}}$, let $\mu_e(R_j)$ denote the number of times string j 's trail traverses edge e (the number of arcs in $\vec{E}(R_j)$ projecting onto e ; values 0, 1, or 2). The homeruns R_j^+ and R_j^- are paths and so traverse each edge at most once. The number of DC cables sharing e is

$$n_e^{\text{DC}} = \sum_{j=1}^{N_{\text{str}}} \left(\mu_e(R_j) + \mathbf{1}[e \in E(R_j^+)] + \mathbf{1}[e \in E(R_j^-)] \right),$$

and the number of DC trenches required on e is

$$\tau_e^{\text{DC}} = \left\lceil \frac{n_e^{\text{DC}}}{N_{\text{DC}}^{\text{trench}}} \right\rceil.$$

With unit trench cost $c_e^{\text{trench}} > 0$, the total DC-trench cost is

$$C_{\text{trench}} = \sum_{e \in E_{\text{used}}^{\text{cross}}} c_e^{\text{trench}} \tau_e^{\text{DC}} w_e.$$

2.6.5 Inverter Cost

Let $c_k^{\text{inv}} > 0$ denote the unit cost of an inverter of type k . The total inverter cost is

$$C_{\text{inv}} = \sum_{(m,k) \in \mathcal{I}} c_k^{\text{inv}}.$$

2.6.6 AC/DC Overlap Penalties

When a DC cross-array edge and an AC edge intersect geometrically, the model permits both to be used but charges a per-pair penalty. The activated overlaps are

$$\mathcal{O}_{\text{used}}^{\text{ov}} = \{ (e_{\text{DC}}, e_{\text{AC}}) \in \mathcal{O}^{\text{cross} \times \text{AC}} \mid e_{\text{DC}} \in E_{\text{used}}^{\text{cross}}, e_{\text{AC}} \in E_{\text{used}}^{\text{AC}} \}.$$

Let $c_{(e_{\text{DC}}, e_{\text{AC}})}^{\text{ov}} > 0$ denote the per-pair penalty. The total overlap cost is

$$C_{\text{ov}} = \sum_{(e_{\text{DC}}, e_{\text{AC}}) \in \mathcal{O}_{\text{used}}^{\text{ov}}} c_{(e_{\text{DC}}, e_{\text{AC}})}^{\text{ov}}.$$

2.6.7 DC-Trench / AC-Internal Overlap Penalties

An overlap occurs at module m when a DC trench crosses through m (cross-array edges on both the H_i^+ and H_i^- sides are used) and both adjacent internal edges carry AC feeders. Each such configuration incurs a penalty. The activated 4-tuples are

$$\mathcal{O}_{\text{used}}^{\text{tov}} = \{ (e^+, e^-, e_1^{\text{int}}, e_2^{\text{int}}) \in \mathcal{O}^{\text{cross} \times \text{int}} \mid e^+, e^- \in E_{\text{used}}^{\text{cross}}, e_1^{\text{int}}, e_2^{\text{int}} \in E_{\text{used}}^{\text{AC}} \}.$$

Let $c_{\omega}^{\text{tov}} > 0$ denote the per-tuple penalty. The total trench-overlap cost is

$$C_{\text{tov}} = \sum_{\omega \in \mathcal{O}_{\text{used}}^{\text{tov}}} c_{\omega}^{\text{tov}}.$$

Chapter 3

Problem modeling with PV-string indices

This chapter formulates the problem of Chapter 2 as a mixed-integer linear program (MILP) with decision variables indexed per PV-string. The feasible-solution tuple of Section 2.5 becomes a system of linear constraints, and the cost function of Section 2.6 becomes the linear objective.

The trails and routes of Section 2.5 are encoded by binary arc-selection variables whose continuity is enforced by degree-balance constraints, balancing the selected arcs entering each vertex against those leaving it. Section 3.1 therefore constructs the directed graphs $\vec{G}^{\text{DC}}$ and $\vec{G}^{\text{AC}}$ on which those constraints are written, and Section 3.2 fixes the derived parameters: capacity bounds, Big-M coefficients, and unit costs. The remaining sections add the constraints in the order of Section 2.5, from inverter placement through to the objective function.

3.1 Graph Transformation

A trail or route of Section 2.5 is traversed in a definite direction, from one terminal to the other, so encoding it by degree-balance constraints—the selected arcs entering a vertex balancing those leaving it—requires directed edges. For $\star \in \{\text{DC}, \text{AC}\}$, we lift each undirected graph $G^\star$ of Section 2.3 to a directed counterpart $\vec{G}^\star$ by replacing every undirected edge with a pair of opposing arcs:

$$\vec{E}^\star = \{ (u, v), (v, u) \mid \{u, v\} \in E^\star \}.$$

At each vertex v , $E_\star^+(v)$ and $E_\star^-(v)$ denote the arcs leaving and entering v in $\vec{G}^\star$, and $\vec{E}^{\text{int}}$, $\vec{E}_{\text{prune}}^{\text{cross}}$ the directed sets induced by E^{int} and $E_{\text{prune}}^{\text{cross}}$.

Since Section 2.4 already removes every edge flagged by $\mathcal{O}^{\text{BND} \times \text{DC}}$ or $\mathcal{O}^{\text{BND} \times \text{AC}}$, both $\vec{G}^\star$ are boundary-safe by construction; no explicit boundary-violation constraint is required.

3.2 Model Parameters

Cost Parameters. The objective function (Section 2.6) uses the unit costs and penalty coefficients below.

Parameter	Description
c_e^{PV}	Unit cost for PV-string cabling on edge e .
c_e^{home}	Unit cost for DC-homerun cabling on edge e .
c_e^{trench}	Unit cost per DC trench on edge e .
c_e^{AC}	Unit cost per AC trench on edge e .
c_k^{inv}	Installation cost of an inverter of type k .
$c_{(e_{\text{DC}}, e_{\text{AC}})}^{\text{ov}}$	Penalty for overlap between DC edge e_{DC} and AC edge e_{AC} .
c_ω^{tov}	Penalty for DC-trench / AC-internal overlap at tuple ω .

Capacity Parameters. Each row in the table below defines a derived bound used as a Big-M coefficient or capacity term in a later constraint.

Symbol	Definition
$Q_{\text{all}}^{\max}$	Longest string length allowed by any inverter type: $Q_{\text{all}}^{\max} = \max_{k \in \mathcal{K}} Q_k^{\max}$.
$Q_{\text{all}}^{\min}$	Longest minimum string length allowed by any inverter type: $Q_{\text{all}}^{\min} = \max_{k \in \mathcal{K}} Q_k^{\min}$.
$I_k^{\max}$	Maximum number of type- k inverters. Each consumes at least $S_k^{\min} Q_k^{\min}$ modules, so: $I_k^{\max} = \left\lfloor \frac{N_{\mathcal{M}}}{S_k^{\min} Q_k^{\min}} \right\rfloor, \quad \forall k \in \mathcal{K}.$
$I_{\text{all}}^{\max}$	Maximum total inverter count, taken across all admissible type choices: $I_{\text{all}}^{\max} = \max_{k \in \mathcal{K}} I_k^{\max}.$
$I_{\text{all}}^{\min}$	Minimum total inverter count required to serve all $N_{\mathcal{M}}$ modules. An inverter of type k accommodates at most $S_k^{\max} Q_k^{\max}$ modules, so: $I_{\text{all}}^{\min} = \left\lceil \frac{N_{\mathcal{M}}}{\max_{k \in \mathcal{K}} (S_k^{\max} Q_k^{\max})} \right\rceil.$
$\Gamma_{\text{DC}}^{\max}$	Maximum number of DC trenches on any single cross-array edge. Each string contributes up to four cables on the edge—two trail traversals (one per arc direction; Section 2.5) and one cable for each of its two homerun directions—so N_{str} strings contribute at most $4 N_{\text{str}}$ cables: $\Gamma_{\text{DC}}^{\max} = \left\lceil \frac{4 N_{\text{str}}}{N_{\text{DC}}^{\text{trench}}} \right\rceil.$

3.3 Decision Variables and Constraints

The decision variables and constraints follow the order of Section 2.5.

3.3.1 Inverter Placement

We introduce binary placement variables:

$y_{k,m} \in \{0,1\}$ Equals 1 if an inverter of type k is installed at module location m ,
 $\forall k \in \mathcal{K}, \forall m \in \mathcal{M}.$

Location Assignment. At most one inverter (of any type) is installed at each module location.

$$\sum_{k \in \mathcal{K}} y_{k,m} \leq 1 \quad \forall m \in \mathcal{M} \quad (\text{C1})$$

A binary indicator tracks which inverter types are used:

$u_k \in \{0,1\}$ Equals 1 if at least one inverter of type k is installed,
 $\forall k \in \mathcal{K}.$

Conditional Activation. The indicator u_k equals 1 exactly when some $y_{k,m} = 1$. The upper bound, with Big-M $I_k^{\max}$, forces $u_k = 1$ once any inverter of type k is placed; the lower bound forces $u_k = 0$ when none is placed, eliminating symmetric solutions in which $u_k = 1$ but the type is unused.

$$u_k \leq \sum_{m \in \mathcal{M}} y_{k,m} \leq I_k^{\max} u_k \quad \forall k \in \mathcal{K} \quad (\text{C2.1})$$

Total Inverter Bound. The total number of installed inverters lies between $I_{\text{all}}^{\min}$ and $I_{\text{all}}^{\max}$ (Section 3.2). Both bounds follow from physical capacity — the upper from minimum per-inverter consumption, the lower from maximum per-inverter coverage — and are stated explicitly to tighten the LP relaxation.

$$I_{\text{all}}^{\min} \leq \sum_{k \in \mathcal{K}} \sum_{m \in \mathcal{M}} y_{k,m} \leq I_{\text{all}}^{\max} \quad (\text{C2.2})$$

Inverter Standardisation. At most $N_{\mathcal{K}}^{\max}$ inverter types are enabled.

$$\sum_{k \in \mathcal{K}} u_k \leq N_{\mathcal{K}}^{\max} \quad (\text{C2.3})$$

3.3.2 PV-Strings

We introduce binary pv-model to string ownership and terminal variables:

$$\begin{aligned} s_{m,j} &\in \{0,1\} && \text{Equals 1 if module } m \text{ is an owned module of PV-string } j, \\ &&& \forall m \in \mathcal{M}, \forall j \in \{1, \dots, N_{\text{str}}\}. \\ b_{m,j}^- &\in \{0,1\} && \text{Equals 1 if module } m \text{ serves as the negative terminal for PV-string } j, \\ &&& \forall m \in \mathcal{M}, \forall j \in \{1, \dots, N_{\text{str}}\}. \\ b_{m,j}^+ &\in \{0,1\} && \text{Equals 1 if module } m \text{ serves as the positive terminal for PV-string } j, \\ &&& \forall m \in \mathcal{M}, \forall j \in \{1, \dots, N_{\text{str}}\}. \end{aligned}$$

Module Partitioning. Every module belongs to exactly one string.

$$\sum_{j=1}^{N_{\text{str}}} s_{m,j} = 1 \quad \forall m \in \mathcal{M} \quad (\text{C3})$$

String Index Symmetry Breaking. The string index j is a label: permuting labels maps a feasible solution to one of equal cost, so every layout appears in up to $N_{\text{str}}!$ copies. Constraint (C3.1) keeps one copy per layout by ordering the strings by their smallest owned module: string j may own module m only if string $j-1$ already owns a module of smaller index.

$$s_{m,j} \leq \sum_{m'=1}^{m-1} s_{m',j-1} \quad \forall m \in \mathcal{M}, \forall j \in \{2, \dots, N_{\text{str}}\} \quad (\text{C3.1})$$

The sum $\sum_{m'=1}^{m-1} s_{m',j-1}$ counts the modules of string $j-1$ that precede m , so $\min\{m : s_{m,j} = 1\}$ increases in j . For $m=1$ that sum is empty and (C3.1) reads $s_{1,j} \leq 0$, which together with (C3) fixes module 1 to string 1 and pushes the empty strings to the highest indices.

Terminal Bounds. Each string has at most one negative and one positive terminal. C6.1 strengthens these inequalities to equalities once the string-to-inverter assignment variable is in place.

$$\sum_{m \in \mathcal{M}} b_{m,j}^- \leq 1, \quad \sum_{m \in \mathcal{M}} b_{m,j}^+ \leq 1 \quad \forall j \in \{1, \dots, N_{\text{str}}\} \quad (\text{C3.2})$$

Terminal Ownership. Terminals are drawn from the owned modules.

$$b_{m,j}^- \leq s_{m,j}, \quad b_{m,j}^+ \leq s_{m,j} \quad \forall m \in \mathcal{M}, \forall j \in \{1, \dots, N_{\text{str}}\} \quad (\text{C3.3})$$

Distinct Terminals. No module serves as both terminals of the same string.

$$b_{m,j}^- + b_{m,j}^+ \leq 1 \quad \forall m \in \mathcal{M}, \forall j \in \{1, \dots, N_{\text{str}}\} \quad (\text{C3.4})$$

3.3.3 Topological Routing

The cabling of each PV-string is a directed trail (Section 2.1) in $\vec{G}^{\text{DC}}$ from the negative terminal to the positive terminal, as specified in Section 2.5: no arc repeats, so each undirected edge $\{u, v\} \in E^{\text{DC}}$ may be traversed at most twice, once in each orientation. The trail visits every owned module; modules with $s_{m,j} = 0$ may appear as extender vertices. We introduce binary arc-selection variables:

$$f_{e,j} \in \{0, 1\} \quad \text{Equals 1 if the trail of PV-string } j \text{ traverses arc } e \text{ (a selection indicator, not a quantity of current),}$$

$$\forall e \in \vec{E}^{\text{DC}}, \forall j \in \{1, \dots, N_{\text{str}}\}.$$

Path Continuity. The selected arcs balance at every module: out-degree minus in-degree is +1 at the negative terminal of string j , -1 at the positive terminal, and 0 elsewhere. This degree condition admits one trail from the negative to the positive terminal plus arc-disjoint closed loops; (C4.3)–(C4.6) exclude the loops.

$$\sum_{e \in E_{\text{DC}}^+(m)} f_{e,j} - \sum_{e \in E_{\text{DC}}^-(m)} f_{e,j} = b_{m,j}^- - b_{m,j}^+ \quad \forall m \in \mathcal{M}, \forall j \in \{1, \dots, N_{\text{str}}\} \quad (\text{C4.1})$$

Module Connectivity. An owned non-terminal module has at least two incident trail edges (enter and leave); an owned terminal has at least one; non-owned modules are unconstrained. The right-hand side of constraint (C4.2) $2s_{m,j} - b_{m,j}^- - b_{m,j}^+$ encodes all three cases. Constraint (C4.2) is implied by the degree balance (C4.1) together with the virtual-flow constraints (C4.3)–(C4.6) stated below; we include it explicitly as a strengthening valid inequality.

$$\sum_{e \in E_{\text{DC}}^+(m)} f_{e,j} + \sum_{e \in E_{\text{DC}}^-(m)} f_{e,j} \geq 2s_{m,j} - b_{m,j}^- - b_{m,j}^+ \quad \forall m \in \mathcal{M}, \forall j \in \{1, \dots, N_{\text{str}}\} \quad (\text{C4.2})$$

Constraints C4.1–C4.2 alone admit *disconnected subtours*: a closed loop of cable edges, disjoint from the source-sink path, satisfies flow conservation while staying isolated from the negative terminal. We rule subtours out with a single-commodity virtual flow that reaches every owned module from the negative terminal. This is the classical *single-commodity flow* (SCF) device introduced by Gavish and Graves [15] for the travelling salesman problem and extended to capacitated tree and routing problems by Gavish [16]. The virtual flow and flow injection are continuous:

$$g_{e,j} \in \mathbb{R}_{\geq 0} \quad \text{Virtual flow on arc } e \text{ for string } j,$$

$$\forall e \in \vec{E}^{\text{DC}}, \forall j \in \{1, \dots, N_{\text{str}}\}.$$

$$\phi_{m,j} \in \mathbb{R}_{\geq 0} \quad \text{Flow injected into the network at negative terminal } b_{m,j}^- \text{ of module } m \text{ for string } j,$$

$$\forall m \in \mathcal{M}, \forall j \in \{1, \dots, N_{\text{str}}\}.$$

Flow–Edge Coupling. Virtual flow travels only on arcs that carry the physical cable. The Big-M $Q_{\text{all}}^{\text{max}}$ bounds the flow on a single arc.

$$g_{e,j} \leq Q_{\text{all}}^{\text{max}} f_{e,j} \quad \forall e \in \vec{E}^{\text{DC}}, \forall j \in \{1, \dots, N_{\text{str}}\} \quad (\text{C4.3})$$

Injection at Negative Terminal. Only the negative terminal of each string injects virtual flow; all other modules inject zero.

$$\phi_{m,j} \leq Q_{\text{all}}^{\text{max}} b_{m,j}^- \quad \forall m \in \mathcal{M}, \forall j \in \{1, \dots, N_{\text{str}}\} \quad (\text{C4.4})$$

Injection Magnitude. Total injection equals the number of modules owned by the string.

$$\sum_{m \in \mathcal{M}} \phi_{m,j} = \sum_{m \in \mathcal{M}} s_{m,j} \quad \forall j \in \{1, \dots, N_{\text{str}}\} \quad (\text{C4.5})$$

Virtual Flow Conservation. Each owned module consumes exactly one unit of virtual flow; each unowned module passes flow through unchanged.

$$\sum_{e \in E_{\text{DC}}^+(m)} g_{e,j} - \sum_{e \in E_{\text{DC}}^-(m)} g_{e,j} = \phi_{m,j} - s_{m,j} \quad \forall m \in \mathcal{M}, \forall j \in \{1, \dots, N_{\text{str}}\} \quad (\text{C4.6})$$

3.3.4 String-to-Inverter Assignments

We introduce binary assignment variables:

$$h_{j,m} \in \{0, 1\} \quad \text{Equals 1 if PV-string } j \text{ is assigned to an inverter at module location } m, \\ \forall j \in \{1, \dots, N_{\text{str}}\}, \forall m \in \mathcal{M}.$$

Unique Assignment. Each string is assigned to at most one inverter.

$$\sum_{m \in \mathcal{M}} h_{j,m} \leq 1 \quad \forall j \in \{1, \dots, N_{\text{str}}\} \quad (\text{C5})$$

Terminal-Assignment Linkage. The terminal bounds tighten to equality: an assigned string has exactly one negative and one positive terminal; an unassigned string has none.

$$\sum_{m \in \mathcal{M}} b_{m,j}^- = \sum_{m \in \mathcal{M}} h_{j,m}, \quad \sum_{m \in \mathcal{M}} b_{m,j}^+ = \sum_{m \in \mathcal{M}} h_{j,m} \quad \forall j \in \{1, \dots, N_{\text{str}}\} \quad (\text{C6.1})$$

String Length. When $h_{j,m} = 1$, the number of owned modules lies within the string-length bounds $[Q_k^{\min}, Q_k^{\max}]$ of the installed type k . When $h_{j,m} = 0$, the Big-M slacks $Q_{\text{all}}^{\max}(1 - h_{j,m})$ and $Q_{\text{all}}^{\min}(1 - h_{j,m})$ deactivate the upper and lower bound, respectively.

$$\sum_{\bar{m} \in \mathcal{M}} s_{\bar{m},j} \leq \sum_{k \in \mathcal{K}} Q_k^{\max} y_{k,m} + Q_{\text{all}}^{\max}(1 - h_{j,m}) \quad \forall j \in \{1, \dots, N_{\text{str}}\}, \forall m \in \mathcal{M} \quad (\text{C6.2})$$

$$\sum_{\bar{m} \in \mathcal{M}} s_{\bar{m},j} \geq \sum_{k \in \mathcal{K}} Q_k^{\min} y_{k,m} - Q_{\text{all}}^{\min}(1 - h_{j,m}) \quad \forall j \in \{1, \dots, N_{\text{str}}\}, \forall m \in \mathcal{M} \quad (\text{C6.3})$$

Assignment Validity. An unassigned string owns no modules; combined with (C6.3), every non-empty string is assigned to some inverter.

$$\sum_{\bar{m} \in \mathcal{M}} s_{\bar{m},j} \leq Q_{\text{all}}^{\max} \sum_{m \in \mathcal{M}} h_{j,m} \quad \forall j \in \{1, \dots, N_{\text{str}}\} \quad (\text{C6.4})$$

Inverter String Capacity. The number of strings assigned to an inverter at location m lies within $[S_k^{\min}, S_k^{\max}]$ for the installed type k . If m has no inverter, both bounds collapse to zero and no string is assigned.

$$\sum_{k \in \mathcal{K}} S_k^{\min} y_{k,m} \leq \sum_{j=1}^{N_{\text{str}}} h_{j,m} \leq \sum_{k \in \mathcal{K}} S_k^{\max} y_{k,m} \quad \forall m \in \mathcal{M} \quad (\text{C6.5})$$

3.3.5 DC-Homeruns

Rather than tracking the positive and negative DC-homerun cables separately (as in Section 2.5), we aggregate them into a single integer count $z_{e,j} \in \{0, 1, 2\}$ on arc e . The aggregation is valid because c_e^{home} is symmetric across the two cables: only the total count on e enters the cost.

$$z_{e,j} \in \{0, 1, 2\} \quad \text{Number of DC-homerun cables of string } j \text{ traversing arc } e, \\ \forall e \in \vec{E}^{\text{DC}}, \forall j \in \{1, \dots, N_{\text{str}}\}.$$

Homerun Flow Conservation. Each present terminal ($b_{m,j}^+ = 1$ or $b_{m,j}^- = 1$) injects one unit of flow; the assigned inverter location ($h_{j,m} = 1$) absorbs both.

$$\sum_{e \in E_{\text{DC}}^+(m)} z_{e,j} - \sum_{e \in E_{\text{DC}}^-(m)} z_{e,j} = b_{m,j}^+ + b_{m,j}^- - 2h_{j,m} \quad \forall m \in \mathcal{M}, \forall j \in \{1, \dots, N_{\text{str}}\} \quad (\text{C7.1})$$

A binary direction selector picks one arc orientation per undirected edge:

$$\delta_{\{u,v\},j}^{\text{home}} \in \{0,1\} \quad \begin{array}{l} \text{Equals 1 if the homerun cabling of string } j \text{ on undirected edge } \{u,v\} \text{ uses the } (u,v) \\ \text{direction; 0 if it uses } (v,u) \text{ or the edge is unused,} \\ \forall \{u,v\} \in E^{\text{DC}}, \forall j \in \{1, \dots, N_{\text{str}}\}. \end{array}$$

Homerun Direction Consistency. The homerun cabling of each string traverses each undirected edge in at most one direction. The coefficient 2, the maximum number of homerun cables per arc, deactivates the unused direction.

$$z_{(u,v),j} \leq 2 \delta_{\{u,v\},j}^{\text{home}} \quad \forall \{u,v\} \in E^{\text{DC}}, \forall j \in \{1, \dots, N_{\text{str}}\} \quad (\text{C7.2})$$

$$z_{(v,u),j} \leq 2(1 - \delta_{\{u,v\},j}^{\text{home}}) \quad \forall \{u,v\} \in E^{\text{DC}}, \forall j \in \{1, \dots, N_{\text{str}}\} \quad (\text{C7.3})$$

3.3.6 DC Circuit Length

The total DC circuit length of a string—the PV-string trail (first sum) plus both homerun cables (second sum)—is at most $L_{\text{DC}}^{\text{max}}$.

$$\sum_{e \in \vec{E}^{\text{DC}}} w_e f_{e,j} + \sum_{e \in \vec{E}^{\text{DC}}} w_e z_{e,j} \leq L_{\text{DC}}^{\text{max}} \quad \forall j \in \{1, \dots, N_{\text{str}}\} \quad (\text{C7.4})$$

3.3.7 Inverter-to-Transformer Assignments

We introduce binary variables for the assignment of inverters to transformers:

$$v_{m,t} \in \{0,1\} \quad \begin{array}{l} \text{Equals 1 if the inverter at location } m \text{ connects to transformer } t, \\ \forall m \in \mathcal{M}, \forall t \in \mathcal{T}. \end{array}$$

Unique Transformer Assignment. Each installed inverter is assigned to exactly one transformer; if no inverter is installed at m , no transformer connection is made.

$$\sum_{t \in \mathcal{T}} v_{m,t} = \sum_{k \in \mathcal{K}} y_{k,m} \quad \forall m \in \mathcal{M} \quad (\text{C8.1})$$

A continuous variable counts the module load each string contributes to its assigned inverter:

$$q_{j,m} \in \mathbb{R}_{\geq 0} \quad \begin{array}{l} \text{Module-load contribution of PV-string } j \text{ to the inverter at location } m, \\ \forall j \in \{1, \dots, N_{\text{str}}\}, \forall m \in \mathcal{M}. \end{array}$$

Inverter Load Aggregation. String j contributes to the inverter at m only when assigned there. The Big-M $Q_{\text{all}}^{\text{max}}$ bounds the contribution.

$$q_{j,m} \leq Q_{\text{all}}^{\text{max}} h_{j,m} \quad \forall j \in \{1, \dots, N_{\text{str}}\}, \forall m \in \mathcal{M} \quad (\text{C9.1})$$

Module Conservation. The contributions of each string across inverter locations sum to its number of owned modules.

$$\sum_{m \in \mathcal{M}} q_{j,m} = \sum_{m \in \mathcal{M}} s_{m,j} \quad \forall j \in \{1, \dots, N_{\text{str}}\} \quad (\text{C9.2})$$

A continuous variable records the load transferred from each inverter to its transformer:

$$\begin{aligned} \ell_{m,t} &\in \mathbb{R}_{\geq 0} && \text{Total module load transferred from inverter } m \text{ to transformer } t, \\ &&& \forall m \in \mathcal{M}, \forall t \in \mathcal{T}. \end{aligned}$$

Transformer Load Coupling. Load transfers from inverter m to transformer t only when $v_{m,t} = 1$. The transformer capacity J_t serves as the Big-M.

$$\ell_{m,t} \leq J_t v_{m,t} \quad \forall m \in \mathcal{M}, \forall t \in \mathcal{T} \quad (\text{C10.1})$$

Inverter Load Balance. The total load leaving an inverter equals the sum of the loads from its assigned strings.

$$\sum_{t \in \mathcal{T}} \ell_{m,t} = \sum_{j=1}^{N_{\text{str}}} q_{j,m} \quad \forall m \in \mathcal{M} \quad (\text{C10.2})$$

Transformer Capacity. At most J_t modules connect to transformer t .

$$\sum_{m \in \mathcal{M}} \ell_{m,t} \leq J_t \quad \forall t \in \mathcal{T} \quad (\text{C11})$$

3.3.8 AC-Feeders

The AC cables route from inverters to transformers as a multi-commodity flow, one commodity per transformer. We introduce integer cable and trench counts and a binary edge-utilisation indicator:

$$\begin{aligned} \alpha_{e,t} &\in \mathbb{Z}_{\geq 0} && \text{Number of AC-feeder cables on arc } e \text{ assigned to transformer } t, \\ &&& \forall e \in \vec{E}^{\text{AC}}, \forall t \in \mathcal{T}. \end{aligned}$$

$$\begin{aligned} \alpha_e^{\text{AC}} &\in \{0, 1\} && \text{Equals 1 if arc } e \text{ is utilised by any AC feeder,} \\ &&& \forall e \in \vec{E}^{\text{AC}}. \end{aligned}$$

$$\begin{aligned} \tau_e^{\text{AC}} &\in \mathbb{Z}_{\geq 0} && \text{Number of AC trenches required on undirected edge } e, \\ &&& \forall e \in E^{\text{AC}}. \end{aligned}$$

AC Flow Conservation. Each inverter assigned to transformer t sources one unit of t -flow; transformer t sinks all of it.

$$\sum_{e \in E_{\text{AC}}^+(u)} \alpha_{e,t} - \sum_{e \in E_{\text{AC}}^-(u)} \alpha_{e,t} = \begin{cases} v_{u,t} & \text{if } u \in \mathcal{M}, \\ -\sum_{m \in \mathcal{M}} v_{m,t} & \text{if } u = t, \\ 0 & \text{otherwise,} \end{cases} \quad \forall t \in \mathcal{T}, \forall u \in V^{\text{AC}} \quad (\text{C12.1})$$

No Transformer Transit. A feeder destined for transformer t avoids every other transformer.

$$\sum_{e \in E_{\text{AC}}^+(t')} \alpha_{e,t} = 0 \quad \forall t, t' \in \mathcal{T}, t' \neq t \quad (\text{C12.2})$$

Edge Activation. An arc is marked used exactly when it carries at least one cable. The Big-M $I_{\text{all}}^{\text{max}}$ bounds the total feeders on a single arc, since each installed inverter routes one feeder.

$$\alpha_e^{\text{AC}} \leq \sum_{t \in \mathcal{T}} \alpha_{e,t} \leq I_{\text{all}}^{\text{max}} \alpha_e^{\text{AC}} \quad \forall e \in \vec{E}^{\text{AC}} \quad (\text{C12.3})$$

AC Edge Direction Consistency. At most one of the two arcs of each undirected AC edge is used. With (C12.3), every AC-feeder cable on the edge shares that direction.

$$\alpha_{(u,v)}^{\text{AC}} + \alpha_{(v,u)}^{\text{AC}} \leq 1 \quad \forall \{u, v\} \in E^{\text{AC}} \quad (\text{C12.4})$$

AC Trench Capacity. Each trench accommodates up to $N_{\text{AC}}^{\text{trench}}$ cables. A physical trench serves both arc directions, so the trench count is defined on undirected edges and aggregates the two arcs.

$$\sum_{t \in \mathcal{T}} (\alpha_{(u,v),t} + \alpha_{(v,u),t}) \leq N_{\text{AC}}^{\text{trench}} \tau_e^{\text{AC}} \quad \forall e = \{u, v\} \in E^{\text{AC}} \quad (\text{C12.5})$$

3.3.9 DC Trenching and Conflicts

A trench count and a use indicator are defined on each cross-array edge:

$$\begin{aligned} \tau_e^{\text{DC}} &\in \mathbb{Z}_{\geq 0} && \text{Number of DC trenches required on edge } e \text{ to accommodate the routed DC cables,} \\ &&& \forall e \in E_{\text{prune}}^{\text{cross}}. \\ \eta_e &\in \{0, 1\} && \text{Equals 1 if any PV-string trail or DC-homerun cable traverses } e, \\ &&& \forall e \in E_{\text{prune}}^{\text{cross}}. \end{aligned}$$

DC Trench Capacity. Each trench accommodates up to $N_{\text{DC}}^{\text{trench}}$ DC cables. The required count on an edge follows from its total cable traversals; a cable that crosses both arc directions of the same undirected edge contributes two units, matching the trench capacity.

$$\sum_{j=1}^{N_{\text{str}}} (f_{(u,v),j} + f_{(v,u),j} + z_{(u,v),j} + z_{(v,u),j}) \leq N_{\text{DC}}^{\text{trench}} \tau_e^{\text{DC}} \quad \forall e = \{u, v\} \in E_{\text{prune}}^{\text{cross}} \quad (\text{C13.1})$$

DC Edge Activation. An edge is marked used exactly when it carries at least one trench. The Big-M $\Gamma_{\text{DC}}^{\text{max}}$ bounds the trench count on any single edge.

$$\eta_e \leq \tau_e^{\text{DC}} \leq \Gamma_{\text{DC}}^{\text{max}} \eta_e \quad \forall e \in E_{\text{prune}}^{\text{cross}} \quad (\text{C13.2})$$

No DC Crossings. For each geometrically conflicting pair $\{e_1, e_2\} \in \mathcal{O}^{\text{cross} \times \text{cross}}$ (Section 2.4), at most one edge is trenched.

$$\eta_{e_1} + \eta_{e_2} \leq 1 \quad \forall \{e_1, e_2\} \in \mathcal{O}^{\text{cross} \times \text{cross}} \quad (\text{C13.3})$$

3.3.10 AC/DC Overlaps

Binary indicators penalised by strictly positive coefficients ($c^{\text{ov}}, c^{\text{tov}} > 0$) flag overlaps. Each constraint below is one-sided, so explicit upper bounds are unnecessary: minimisation drives each indicator to 0 unless its constraint pins it to 1.

$$o_{e^{\text{DC}}, e^{\text{AC}}} \in \{0, 1\} \quad \text{Equals 1 when DC cross-array edge } e^{\text{DC}} \text{ and AC edge } e^{\text{AC}} \text{ are simultaneously used,} \\ \forall (e^{\text{DC}}, e^{\text{AC}}) \in \mathcal{O}^{\text{cross} \times \text{AC}}.$$

AC/DC overlap penalties. The indicator is forced to 1 when DC trench e^{DC} is used and AC edge $e^{\text{AC}} = \{u, v\}$ carries flow in either direction. By (C12.4), $\alpha_{(u,v)}^{\text{AC}} + \alpha_{(v,u)}^{\text{AC}} \in \{0, 1\}$, so the sum is the undirected-edge utilisation indicator.

$$o_{e^{\text{DC}}, e^{\text{AC}}} \geq \eta_{e^{\text{DC}}} + \alpha_{(u,v)}^{\text{AC}} + \alpha_{(v,u)}^{\text{AC}} - 1 \quad \forall (e^{\text{DC}}, e^{\text{AC}} = \{u, v\}) \in \mathcal{O}^{\text{cross} \times \text{AC}} \quad (\text{C14})$$

DC-trench / AC-internal overlap penalties. A penalty applies when a DC trench crosses a module on both sides and both adjacent internal edges carry AC feeders (Section 2.4). One binary indicator is introduced per 4-tuple:

$$o_\omega \in \{0, 1\} \quad \text{Equals 1 if the DC trench and AC cables overlap at the module identified by the 4-tuple } \omega, \\ \forall \omega \in \mathcal{O}^{\text{cross} \times \text{int}}.$$

For each tuple $\omega = (e^+, e^-, e_1^{\text{int}} = \{u_1, v_1\}, e_2^{\text{int}} = \{u_2, v_2\})$, the indicator is forced to 1 when both cross-array edges carry DC trenches and both internal edges carry AC flow. As above, (C12.4) lets each internal-edge utilisation appear as the sum of its two arc directions.

$$o_\omega \geq \eta_{e^+} + \eta_{e^-} + \alpha_{(u_1, v_1)}^{\text{AC}} + \alpha_{(v_1, u_1)}^{\text{AC}} + \alpha_{(u_2, v_2)}^{\text{AC}} + \alpha_{(v_2, u_2)}^{\text{AC}} - 3 \quad \forall \omega \in \mathcal{O}^{\text{cross} \times \text{int}} \quad (\text{C15})$$

3.3.11 Objective Function

The MILP minimises total layout cost: the cost of cabling, trenching, and inverters plus penalty terms for geometric overlaps.

$$\begin{aligned} \min Z = & \underbrace{\sum_{e \in \vec{E}^{\text{DC}}} \sum_{j=1}^{N_{\text{str}}} c_e^{\text{pv}} f_{e,j} w_e}_{\text{PV-string edges}} + \underbrace{\sum_{e \in \vec{E}^{\text{DC}}} \sum_{j=1}^{N_{\text{str}}} c_e^{\text{home}} z_{e,j} w_e}_{\text{DC-homerun edges}} \\ & + \underbrace{\sum_{e \in E^{\text{AC}}} c_e^{\text{AC}} \tau_e^{\text{AC}} w_e}_{\text{AC-feeder trenches}} + \underbrace{\sum_{e \in E_{\text{prune}}^{\text{cross}}} c_e^{\text{trench}} \tau_e^{\text{DC}} w_e}_{\text{DC-trench cost}} + \underbrace{\sum_{m \in \mathcal{M}} \sum_{k \in \mathcal{K}} c_k^{\text{inv}} y_{k,m}}_{\text{Inverter cost}} \\ & + \underbrace{\sum_{(e^{\text{DC}}, e^{\text{AC}}) \in \mathcal{O}^{\text{cross} \times \text{AC}}} c_{(e^{\text{DC}}, e^{\text{AC}})}^{\text{ov}} o_{e^{\text{DC}}, e^{\text{AC}}}}_{\text{AC/DC overlap penalties}} + \underbrace{\sum_{\omega \in \mathcal{O}^{\text{cross} \times \text{int}}} c_\omega^{\text{tov}} o_\omega}_{\text{Trench/AC overlap penalties}} \end{aligned} \quad (\text{OBJ})$$

Chapter 4

Problem modeling with inverter type indices

The string-indexed model of Chapter 3 carries one set of decision variables per PV-string, so its size grows with the number of modules. This chapter aggregates strings by inverter type, reducing the model to a size that depends only on $|\mathcal{K}|$. The Graph Transformation and Model Parameters of Chapter 3 are reused unchanged.

Aggregation trades away exact circuit-loop reconstruction and string-level edge ownership; in their place the model recovers an averaged DC circuit length (Constraint (C8.3)) and a per-edge accumulated-distance bound (Constraint (C8.4)). The sections that follow add the constraints in the order of a feasible solution, from inverter placement through to the objective function.

4.1 Decision Variables and Constraints

The decision variables and constraints follow the order of Section 2.5.

4.1.1 Inverter Placement

We introduce binary decision variables for inverter installation:

$y_{k,m} \in \{0,1\}$ Equals 1 if an inverter of type k is installed at module location m ,
 $\forall k \in \mathcal{K}, \forall m \in \mathcal{M}$.

Location Assignment. At most one inverter (of any type) is installed at each module location.

$$\sum_{k \in \mathcal{K}} y_{k,m} \leq 1 \quad \forall m \in \mathcal{M} \quad (\text{C1})$$

A binary indicator tracks which inverter types are used:

$u_k \in \{0,1\}$ Equals 1 if at least one inverter of type k is installed,
 $\forall k \in \mathcal{K}$.

Conditional Activation. The indicator u_k equals 1 exactly when some $y_{k,m} = 1$. The upper bound, with Big-M $I_k^{\max}$, forces $u_k = 1$ once any inverter of type k is placed; the lower bound forces $u_k = 0$ when none is placed, eliminating symmetric solutions in which $u_k = 1$ but the type is unused.

$$u_k \leq \sum_{m \in \mathcal{M}} y_{k,m} \leq I_k^{\max} u_k \quad \forall k \in \mathcal{K} \quad (\text{C2.1})$$

Total Inverter Bound. The total number of installed inverters lies between $I_{\text{all}}^{\min}$ and $I_{\text{all}}^{\max}$ (Section 3.2). Both bounds follow from physical capacity — the upper from minimum per-inverter consumption, the lower from maximum per-inverter coverage — and are stated explicitly to tighten the LP relaxation.

$$I_{\text{all}}^{\min} \leq \sum_{k \in \mathcal{K}} \sum_{m \in \mathcal{M}} y_{k,m} \leq I_{\text{all}}^{\max} \quad (\text{C2.2})$$

Inverter Standardisation. At most $N_{\mathcal{K}}^{\max}$ inverter types are enabled.

$$\sum_{k \in \mathcal{K}} u_k \leq N_{\mathcal{K}}^{\max} \quad (\text{C2.3})$$

4.1.2 PV-Strings

Rather than tracking unique string identities as in Chapter 3, Chapter 4 aggregates strings by inverter type. We introduce ownership and terminal variables at the inverter type level:

$$\begin{aligned} s_{m,k} &\in \{0,1\} && \text{Equals 1 if module } m \text{ is owned by a string of inverter type } k, \\ &&& \forall m \in \mathcal{M}, \forall k \in \mathcal{K}. \\ b_{m,k}^- &\in \{0,1\} && \text{Equals 1 if module } m \text{ serves as a negative terminal for a string of inverter type } k, \\ &&& \forall m \in \mathcal{M}, \forall k \in \mathcal{K}. \\ b_{m,k}^+ &\in \{0,1\} && \text{Equals 1 if module } m \text{ serves as a positive terminal for a string of inverter type } k, \\ &&& \forall m \in \mathcal{M}, \forall k \in \mathcal{K}. \end{aligned}$$

Module Partitioning. Every module is owned by exactly one inverter type.

$$\sum_{k \in \mathcal{K}} s_{m,k} = 1 \quad \forall m \in \mathcal{M} \quad (\text{C4.5})$$

Terminal Balance. Every string has exactly one negative and one positive terminal.

$$\sum_{m \in \mathcal{M}} b_{m,k}^- = \sum_{m \in \mathcal{M}} b_{m,k}^+ \quad \forall k \in \mathcal{K} \quad (\text{C3})$$

Distinct Terminals. A module cannot simultaneously serve as both the negative and positive terminal for the same inverter type.

$$b_{m,k}^- + b_{m,k}^+ \leq 1 \quad \forall m \in \mathcal{M}, \forall k \in \mathcal{K} \quad (\text{C3.1})$$

Terminal Ownership. The negative and positive terminals of type k must belong to modules owned by that same type.

$$b_{m,k}^- \leq s_{m,k}, \quad b_{m,k}^+ \leq s_{m,k} \quad \forall m \in \mathcal{M}, \forall k \in \mathcal{K} \quad (\text{C3.2})$$

4.1.3 Topological Routing

We model PV-string cabling as integer flows on $\vec{G}^{\text{DC}}$. The variable $f_{e,k}$ counts type- k strings on each arc e .

$$\begin{aligned} f_{e,k} &\in \mathbb{Z}_{\geq 0} && \text{Number of PV-strings of type } k \text{ traversing arc } e, \\ &&& \forall e \in \vec{E}^{\text{DC}}, \forall k \in \mathcal{K}. \end{aligned}$$

String Flow Conservation. The negative terminals $b_{m,k}^-$ act as sources, each injecting one unit of string flow; the positive terminals $b_{m,k}^+$ act as sinks, each absorbing one unit.

$$\sum_{e \in E_{\text{DC}}^+(m)} f_{e,k} - \sum_{e \in E_{\text{DC}}^-(m)} f_{e,k} = b_{m,k}^- - b_{m,k}^+ \quad \forall m \in \mathcal{M}, \forall k \in \mathcal{K} \quad (\text{C4.1})$$

Flow conservation alone leaves two pathologies open: a type- k string may exceed its length limit $Q_k^{\max}$, and a closed loop of $f_{e,k}$ disjoint from the terminals can satisfy conservation while carrying no owned modules. We address both with a cumulative module-load flow $g_{e,k}$ that picks up one unit per owned module and discharges it at the positive terminal. This is the classical *single-commodity flow* (SCF) device introduced by Gavish and Graves [15] for the travelling salesman problem and extended to capacitated tree and routing problems by Gavish [16].

$g_{e,k} \in \mathbb{Z}_{\geq 0}$ Cumulative number of owned modules carried by type- k strings on arc e ,
 $\forall e \in \vec{E}^{\text{DC}}, \forall k \in \mathcal{K}$.

$q_{m,k}^{\text{dump}} \in \mathbb{R}_{\geq 0}$ Module load discharged at module m when it serves as a positive terminal of type k ,
 $\forall m \in \mathcal{M}, \forall k \in \mathcal{K}$.

Flow–Edge Coupling. The module-load flow may travel only on arcs that carry physical cable. The Big-M coefficient $Q_k^{\max}$ tightly bounds the load each type- k string can carry.

$$g_{e,k} \leq Q_k^{\max} f_{e,k} \quad \forall e \in \vec{E}^{\text{DC}}, \forall k \in \mathcal{K} \quad (\text{C4.2})$$

String Length. At any positive terminal ($b_{m,k}^+ = 1$), the discharged load equals the length of the string terminating there and must fall within $[Q_k^{\min}, Q_k^{\max}]$. When $b_{m,k}^+ = 0$, both bounds collapse to zero, forcing $q_{m,k}^{\text{dump}}$ to vanish.

$$Q_k^{\min} b_{m,k}^+ \leq q_{m,k}^{\text{dump}} \leq Q_k^{\max} b_{m,k}^+ \quad \forall m \in \mathcal{M}, \forall k \in \mathcal{K} \quad (\text{C4.3})$$

Module Accumulation. Each owned module ($s_{m,k} = 1$) injects one unit of load into the network; positive terminals absorb the accumulated load into $q_{m,k}^{\text{dump}}$. Combined with the Flow–Edge Coupling, the construction forces every owned module to lie on a path of nonzero $f_{e,k}$ that reaches a positive terminal, ruling out the disconnected subtour described above.

$$\sum_{e \in E_{\text{DC}}^+(m)} g_{e,k} - \sum_{e \in E_{\text{DC}}^-(m)} g_{e,k} = s_{m,k} - q_{m,k}^{\text{dump}} \quad \forall m \in \mathcal{M}, \forall k \in \mathcal{K} \quad (\text{C4.4})$$

4.1.4 String-to-Inverter Assignments

We introduce an integer variable for the per-type string count at each inverter location:

$h_{m,k} \in \mathbb{Z}_{\geq 0}$ Number of PV-strings of type k assigned to the inverter at module location m ,
 $\forall m \in \mathcal{M}, \forall k \in \mathcal{K}$.

Inverter String Capacity. If an inverter of type k is installed at location m ($y_{k,m} = 1$), the number of assigned strings must lie in $[S_k^{\min}, S_k^{\max}]$. If no inverter of type k is installed ($y_{k,m} = 0$), both bounds collapse to zero.

$$S_k^{\min} y_{k,m} \leq h_{m,k} \leq S_k^{\max} y_{k,m} \quad \forall m \in \mathcal{M}, \forall k \in \mathcal{K} \quad (\text{C5})$$

An aggregate variable totals the string count at each inverter location:

$h_m \in \mathbb{R}_{\geq 0}$ Aggregate number of PV-strings assigned to the inverter at module location m ,
 $\forall m \in \mathcal{M}$.

String Aggregation. The aggregate h_m collects the per-type counts.

$$h_m = \sum_{k \in \mathcal{K}} h_{m,k} \quad \forall m \in \mathcal{M} \quad (\text{C6})$$

4.1.5 DC-Homeruns

We route the positive and negative DC-homerun cables from each string's terminals to its assigned inverter, indexed by inverter type k so that a cable can serve only a matching inverter. The negative homerun is modelled in the *reverse* physical direction — from the inverter back to the negative terminal — so that the three cable segments form a directed circuit:

$$\text{Inverter} \xrightarrow{z^-} \text{Negative Terminal} \xrightarrow{f} \text{Positive Terminal} \xrightarrow{z^+} \text{Inverter}.$$

The DC circuit length formulation in the next section exploits this structure.

We introduce integer variables for the positive and negative homerun flows:

$$\begin{aligned} z_{e,k}^+ &\in \mathbb{Z}_{\geq 0} && \text{Number of positive DC-homerun cables of type } k \text{ on edge } e, \\ &&& \forall e \in \vec{E}^{\text{DC}}, \forall k \in \mathcal{K}. \\ z_{e,k}^- &\in \mathbb{Z}_{\geq 0} && \text{Number of negative DC-homerun cables of type } k \text{ on edge } e \text{ (reverse direction),} \\ &&& \forall e \in \vec{E}^{\text{DC}}, \forall k \in \mathcal{K}. \end{aligned}$$

Positive Homerun Flow Conservation. At each module m , the positive-terminal indicator $b_{m,k}^+$ sources one unit of positive-homerun flow; the inverter at m sinks one unit per assigned type- k string, for a total of $h_{m,k}$ units.

$$\sum_{e \in E_{\text{DC}}^+(m)} z_{e,k}^+ - \sum_{e \in E_{\text{DC}}^-(m)} z_{e,k}^+ = b_{m,k}^+ - h_{m,k} \quad \forall m \in \mathcal{M}, \forall k \in \mathcal{K} \quad (\text{C7.1})$$

Negative Homerun Flow Conservation. The roles reverse for the negative homerun: the inverter at m sources $h_{m,k}$ units, and the negative-terminal indicator $b_{m,k}^-$ sinks one. This closes the directed circuit that originates and terminates at the inverter.

$$\sum_{e \in E_{\text{DC}}^+(m)} z_{e,k}^- - \sum_{e \in E_{\text{DC}}^-(m)} z_{e,k}^- = h_{m,k} - b_{m,k}^- \quad \forall m \in \mathcal{M}, \forall k \in \mathcal{K} \quad (\text{C7.2})$$

C7.1 and C7.2 force the number of type- k strings $h_{m,k}$ to be tied to the actual string-to-inverter assignments. Both constraints balance *counts* rather than identities: each inverter receives as many positive homeruns as it emits negative ones, but because $z_{e,k}^\pm$ carries no string index, a decomposition of the flow into individual circuits may route a string's positive homerun to one inverter and its negative homerun to another.

4.1.6 DC Circuit Length

The DC circuit length limit $L_{\text{DC}}^{\max}$ caps the total cable length of each string (PV-string trail plus both homeruns) to bound the voltage drop. Because $f_{e,k}$ and $z_{e,k}^\pm$ track string counts and not string identities, this formulation cannot impose the limit on each individual string. We instead employ a *flow-moment* construction that bounds the *average* DC circuit length across all strings assigned to each inverter, supplemented by a per-edge accumulated-distance bound that tightens the LP relaxation. The averaging is a deliberate relaxation: individual strings may still exceed $L_{\text{DC}}^{\max}$ when they share edges with shorter circuits, but the per-edge bound narrows the gap.

The construction uses the directed circuit of the previous section. We aggregate the typed flows into a circulation variable:

$$\begin{aligned} \phi_e &\in \mathbb{R}_{\geq 0} && \text{Total number of DC cable routes (string cables and both homeruns) on edge } e, \\ &&& \forall e \in \vec{E}^{\text{DC}}. \end{aligned}$$

Flow Aggregation. The circulation ϕ_e sums the string flow and both homerun flows over all types:

$$\phi_e = \sum_{k \in \mathcal{K}} (f_{e,k} + z_{e,k}^+ + z_{e,k}^-) \quad \forall e \in \vec{E}^{\text{DC}} \quad (\text{C8.1})$$

We introduce continuous variables for the cumulative distance mass and its absorption at inverter locations:

$$\rho_e \in \mathbb{R}_{\geq 0} \quad \begin{array}{l} \text{Cumulative distance mass on edge } e, \\ \forall e \in \vec{E}^{\text{DC}}. \end{array}$$

$$\Delta_m \in \mathbb{R}_{\geq 0} \quad \begin{array}{l} \text{Total distance mass absorbed by the inverter at location } m, \\ \forall m \in \mathcal{M}. \end{array}$$

Distance Mass Conservation. As the circulation traverses an edge e , it accumulates distance mass proportional to the edge weight w_e . At each inverter location, the total accumulated circuit distance is absorbed into Δ_m .

$$\sum_{e \in E_{\text{DC}}^+(m)} \rho_e - \sum_{e \in E_{\text{DC}}^-(m)} \rho_e = \sum_{e \in E_{\text{DC}}^-(m)} w_e \phi_e - \Delta_m \quad \forall m \in \mathcal{M} \quad (\text{C8.2})$$

Average Circuit Length Bound. The total distance mass at each inverter must not exceed the DC circuit limit $L_{\text{DC}}^{\max}$ multiplied by the number of strings h_m assigned to that inverter. This enforces that the average DC circuit length per string satisfies the voltage-drop limit:

$$\Delta_m \leq L_{\text{DC}}^{\max} h_m \quad \forall m \in \mathcal{M} \quad (\text{C8.3})$$

Per-Edge Distance Bound. Constraint (C8.3) binds the accumulated distance only at the inverter. Since each unit of circulation carries one string's directed circuit and no individual circuit may exceed $L_{\text{DC}}^{\max}$, the same bound holds at every edge:

$$\rho_e \leq L_{\text{DC}}^{\max} \phi_e \quad \forall e \in \vec{E}^{\text{DC}} \quad (\text{C8.4})$$

4.1.7 Inverter-to-Transformer Assignments

Each installed inverter connects to exactly one transformer. The section proceeds in three steps: assign inverters to transformers, derive each inverter's PV-module load from the ownership variables, and transfer the load subject to transformer capacity.

We introduce a binary variable for the inverter-to-transformer assignment:

$$v_{m,t} \in \{0, 1\} \quad \begin{array}{l} \text{Equals 1 if the inverter at location } m \text{ connects to transformer } t, \\ \forall m \in \mathcal{M}, \forall t \in \mathcal{T}. \end{array}$$

Transformer Assignment. An installed inverter at location m must be assigned to exactly one transformer. If no inverter is installed ($\sum_k y_{k,m} = 0$), no assignment is permitted.

$$\sum_{t \in \mathcal{T}} v_{m,t} = \sum_{k \in \mathcal{K}} y_{k,m} \quad \forall m \in \mathcal{M} \quad (\text{C9.1})$$

We introduce a per-type load variable:

$$L_{m,k} \in \mathbb{R}_{\geq 0} \quad \begin{array}{l} \text{Total PV-module load delivered to a type-} k \text{ inverter at location } m, \\ \forall m \in \mathcal{M}, \forall k \in \mathcal{K}. \end{array}$$

Total Load Conservation. Each type's total delivered load equals the number of modules it owns.

$$\sum_{m \in \mathcal{M}} L_{m,k} = \sum_{m \in \mathcal{M}} s_{m,k} \quad \forall k \in \mathcal{K} \quad (\text{C10.1})$$

Inverter Load Consistency. The load delivered to inverter (m, k) lies between $Q_k^{\min} h_{m,k}$ and $Q_k^{\max} h_{m,k}$, the string-length bounds scaled by the assigned string count.

$$Q_k^{\min} h_{m,k} \leq L_{m,k} \leq Q_k^{\max} h_{m,k} \quad \forall m \in \mathcal{M}, \forall k \in \mathcal{K} \quad (\text{C10.2})$$

The aggregate load L_m sums over types:

$$L_m \in \mathbb{R}_{\geq 0} \quad \text{Total PV-module load at the inverter at location } m, \\ \forall m \in \mathcal{M}.$$

Load Aggregation. Sum the per-type loads:

$$L_m = \sum_{k \in \mathcal{K}} L_{m,k} \quad \forall m \in \mathcal{M} \quad (\text{C10.3})$$

We introduce a continuous variable for the inverter-to-transformer load transfer:

$$\ell_{m,t} \in \mathbb{R}_{\geq 0} \quad \text{Total module load transferred from inverter } m \text{ to transformer } t, \\ \forall m \in \mathcal{M}, \forall t \in \mathcal{T}.$$

Load Flow Balance. Conserve module load at each inverter.

$$\sum_{t \in \mathcal{T}} \ell_{m,t} = L_m \quad \forall m \in \mathcal{M} \quad (\text{C10.4})$$

Conditional Load Transfer. Load transfer from inverter m to transformer t is permitted only if the assignment exists ($v_{m,t} = 1$). The Big-M coefficient is strengthened to $\min(J_t, \max_{k \in \mathcal{K}} Q_k^{\max} S_k^{\max})$, which takes the smaller of the transformer capacity and the maximum module load any single inverter can serve:

$$\ell_{m,t} \leq \min(J_t, \max_{k \in \mathcal{K}} (Q_k^{\max} S_k^{\max})) v_{m,t} \quad \forall m \in \mathcal{M}, \forall t \in \mathcal{T} \quad (\text{C11.1})$$

Transformer Capacity. Each transformer respects its capacity J_t :

$$\sum_{m \in \mathcal{M}} \ell_{m,t} \leq J_t \quad \forall t \in \mathcal{T} \quad (\text{C11.2})$$

4.1.8 AC-Feeders

AC cables route from inverters to transformers as a multi-commodity flow on $\vec{G}^{\text{AC}}$, with one commodity per transformer. Three variables suffice: a per-commodity cable count, an arc-utilisation indicator, and an undirected trench count.

$$\alpha_{e,t} \in \mathbb{Z}_{\geq 0} \quad \text{Number of AC-feeder cables on arc } e \text{ routed to transformer } t, \\ \forall e \in \vec{E}^{\text{AC}}, \forall t \in \mathcal{T}.$$

$$\alpha_e^{\text{AC}} \in \{0, 1\} \quad \text{Equals 1 if arc } e \text{ is utilised by any AC feeder,} \\ \forall e \in \vec{E}^{\text{AC}}.$$

$$\tau_e^{\text{AC}} \in \mathbb{Z}_{\geq 0} \quad \text{Number of AC trenches required on undirected edge } e, \\ \forall e \in E^{\text{AC}}.$$

AC Flow Conservation. For each commodity t , every inverter assigned to t (i.e., $v_{m,t} = 1$) sources one unit of t -flow; transformer t sinks all sourced flow; all other vertices act as pure transit.

$$\sum_{e \in E_{\text{AC}}^+(u)} \alpha_{e,t} - \sum_{e \in E_{\text{AC}}^-(u)} \alpha_{e,t} = \begin{cases} v_{u,t} & \text{if } u \in \mathcal{M}, \\ -\sum_{m \in \mathcal{M}} v_{m,t} & \text{if } u = t, \\ 0 & \text{otherwise,} \end{cases} \quad \forall t \in \mathcal{T}, \forall u \in V^{\text{AC}} \quad (\text{C12.1})$$

No Transformer Transit. A feeder routed to transformer t must not pass through any other transformer $t' \neq t$.

$$\sum_{e \in E_{AC}^+(t')} \alpha_{e,t} = 0 \quad \forall t, t' \in \mathcal{T}, t' \neq t \quad (C12.2)$$

Edge Activation. An arc is marked used if and only if it carries at least one cable. The Big-M coefficient $I_{all}^{\max}$ tightly bounds the total number of AC feeders on a single arc, since at most one feeder is routed per installed inverter.

$$\alpha_e^{AC} \leq \sum_{t \in \mathcal{T}} \alpha_{e,t} \leq I_{all}^{\max} \alpha_e^{AC} \quad \forall e \in \vec{E}^{AC} \quad (C12.3)$$

AC Edge Direction Consistency. On each undirected AC edge, at most one of the two arc directions can be utilised. Combined with the Edge Activation bound, the restriction forces every AC feeder on the edge to share that direction.

$$\alpha_{(u,v)}^{AC} + \alpha_{(v,u)}^{AC} \leq 1 \quad \forall \{u, v\} \in E^{AC} \quad (C12.4)$$

AC Trench Capacity. Multiple AC-feeder cables share a single trench, with each trench accommodating up to N_{AC}^{trench} cables. Because a physical trench serves cables in both arc directions, the trench count is defined on undirected edges and aggregates both directions.

$$\sum_{t \in \mathcal{T}} (\alpha_{(u,v),t} + \alpha_{(v,u),t}) \leq N_{AC}^{\text{trench}} \tau_e^{AC} \quad \forall e = \{u, v\} \in E^{AC} \quad (C12.5)$$

4.1.9 DC Trenching and Conflicts

Each trench accommodates up to N_{DC}^{trench} DC cables. We introduce a trench count and an edge-utilisation indicator on each cross-array edge:

$$\begin{aligned} \tau_e^{DC} &\in \mathbb{Z}_{\geq 0} && \text{Number of DC trenches required on edge } e, \\ &&& \forall e \in E_{\text{prune}}^{\text{cross}}. \\ \eta_e &\in \{0, 1\} && \text{Equals 1 if edge } e \text{ is utilised by any DC cable,} \\ &&& \forall e \in E_{\text{prune}}^{\text{cross}}. \end{aligned}$$

DC Trench Capacity. The trench count on each undirected edge absorbs both arc directions of the aggregated cable flow ϕ_e (Constraint (C8.1)); an edge traversed both ways contributes two cables to the trench requirement.

$$\phi_{(u,v)} + \phi_{(v,u)} \leq N_{DC}^{\text{trench}} \tau_e^{DC} \quad \forall e = \{u, v\} \in E_{\text{prune}}^{\text{cross}} \quad (C13.1)$$

DC Edge Activation. An edge is marked used if and only if it carries at least one DC trench. The Big-M coefficient $\Gamma_{DC}^{\max}$ tightly bounds the number of DC trenches on any single edge.

$$\eta_e \leq \tau_e^{DC} \leq \Gamma_{DC}^{\max} \eta_e \quad \forall e \in E_{\text{prune}}^{\text{cross}} \quad (C13.2)$$

No DC Crossings. For each pair of geometrically conflicting cross-array edges, at most one may be used.

$$\eta_{e_1} + \eta_{e_2} \leq 1 \quad \forall \{e_1, e_2\} \in \mathcal{O}^{\text{cross} \times \text{cross}} \quad (C13.3)$$

4.1.10 AC/DC Overlaps

The objective penalises the binary indicators below with strictly positive coefficients ($c^{\text{ov}}, c^{\text{tov}} > 0$). Cost minimisation then drives each indicator to 0 unless its forcing inequality pins it to 1; no explicit upper bound is needed.

$$o_{e^{\text{DC}}, e^{\text{AC}}} \in \{0, 1\} \quad \text{Equals 1 if the DC trench } e^{\text{DC}} \text{ and the AC edge } e^{\text{AC}} \text{ are simultaneously used,} \\ \forall (e^{\text{DC}}, e^{\text{AC}}) \in \mathcal{O}^{\text{cross} \times \text{AC}}.$$

AC/DC Overlap Penalties. The indicator $o_{e^{\text{DC}}, e^{\text{AC}}}$ is forced to 1 when the DC trench e^{DC} is used and the AC edge $e^{\text{AC}} = \{u, v\}$ carries flow in either direction. Constraint (C12.4) ensures $\alpha_{(u,v)}^{\text{AC}} + \alpha_{(v,u)}^{\text{AC}} \in \{0, 1\}$, so the sum acts as the undirected-edge utilisation indicator.

$$o_{e^{\text{DC}}, e^{\text{AC}}} \geq \eta_{e^{\text{DC}}} + \alpha_{(u,v)}^{\text{AC}} + \alpha_{(v,u)}^{\text{AC}} - 1 \quad \forall (e^{\text{DC}}, e^{\text{AC}} = \{u, v\}) \in \mathcal{O}^{\text{cross} \times \text{AC}} \quad (\text{C14})$$

DC-Trench / AC-Internal Overlap Penalties. When a DC trench crosses a module on both sides and both adjacent internal edges carry AC feeders (Section 2.4), an overlap penalty applies. We introduce a binary indicator per 4-tuple:

$$o_\omega \in \{0, 1\} \quad \text{Equals 1 if the DC trench and AC cables overlap at the module identified by 4-tuple } \omega, \\ \forall \omega \in \mathcal{O}^{\text{cross} \times \text{int}}.$$

For each 4-tuple $\omega = (e^+, e^-, e_1^{\text{int}} = \{u_1, v_1\}, e_2^{\text{int}} = \{u_2, v_2\})$, the indicator is forced to 1 when both cross-array edges carry DC trenches and both internal edges carry AC flow. By Constraint (C12.4), each internal-edge utilisation equals the sum of its two arc directions.

$$o_\omega \geq \eta_{e^+} + \eta_{e^-} + \alpha_{(u_1, v_1)}^{\text{AC}} + \alpha_{(v_1, u_1)}^{\text{AC}} + \alpha_{(u_2, v_2)}^{\text{AC}} + \alpha_{(v_2, u_2)}^{\text{AC}} - 3 \quad \forall \omega \in \mathcal{O}^{\text{cross} \times \text{int}} \quad (\text{C15})$$

4.1.11 Objective Function

The MILP minimises total layout cost: the cost of cabling, trenching, and inverters, plus penalty terms for geometric overlaps.

$$\begin{aligned} \min Z = & \underbrace{\sum_{e \in \vec{E}^{\text{DC}}} \sum_{k \in \mathcal{K}} c_e^{\text{PV}} f_{e,k} w_e}_{\text{PV-string edges}} + \underbrace{\sum_{e \in \vec{E}^{\text{DC}}} \sum_{k \in \mathcal{K}} c_e^{\text{home}} (z_{e,k}^+ + z_{e,k}^-) w_e}_{\text{DC-homerun edges}} \\ & + \underbrace{\sum_{e \in E^{\text{AC}}} c_e^{\text{AC}} \tau_e^{\text{AC}} w_e}_{\text{AC-feeder trenches}} + \underbrace{\sum_{e \in E_{\text{prune}}^{\text{cross}}} c_e^{\text{trench}} \tau_e^{\text{DC}} w_e}_{\text{DC-trench cost}} + \underbrace{\sum_{m \in \mathcal{M}} \sum_{k \in \mathcal{K}} c_k^{\text{inv}} y_{k,m}}_{\text{Inverter cost}} \\ & + \underbrace{\sum_{(e^{\text{DC}}, e^{\text{AC}}) \in \mathcal{O}^{\text{cross} \times \text{AC}}} c_{(e^{\text{DC}}, e^{\text{AC}})}^{\text{ov}} o_{e^{\text{DC}}, e^{\text{AC}}}}_{\text{AC/DC overlap penalties}} + \underbrace{\sum_{\omega \in \mathcal{O}^{\text{cross} \times \text{int}}} c_\omega^{\text{tov}} o_\omega}_{\text{Trench/AC overlap penalties}} \end{aligned} \quad (\text{OBJ})$$

Chapter 5

Monolithic Solver Evaluation

The string-indexed model of Chapter 3 and the inverter-type-indexed model of Chapter 4 both express the joint inverter-placement, string-partitioning, and cable-routing problem as a single mixed-integer linear program. Writing the models down says nothing about whether either can actually be solved on real utility-scale layouts; that has to be measured, and this chapter measures it through a controlled computational study.

Two questions set up the experiments. The first concerns scaling: how do the size of each monolithic formulation and the time needed to solve it grow with the instance size, and which structural feature of an instance dominates solver effort? The second concerns diagnosis: on real plant layouts, which class of decision variables is the tractability bottleneck, in the sense that fixing or hinting it a priori would most reduce the solver’s effort?

Section 5.1 introduces the hardware, the solver, and the cost coefficients held constant across all experiments. Section 5.2 reports the scaling study on synthetic instances, and Section 5.3 the decision-freezing study on nine real plants.

5.1 Experimental Setup

All experiments ran on a single workstation: an AMD Ryzen 5 7600X (Zen 4; 6 physical cores, 12 logical threads, 4.7 GHz base / 5.3 GHz boost) with 32 GB of RAM under Windows 11 Pro. The solver is Gurobi 13.0.2, invoked through `gurobipy`, with every setting at its default: all 12 threads, a relative MIP-gap target of 10^{-4} (0.01%), and presolve, symmetry handling, cuts, and heuristics unchanged.

Coefficient	Value	Role
c_e^{pv}	1.0	PV-string cabling, per metre
c_e^{home}	0.1	DC-homerun cabling, per metre
c_e^{trench}	50.0	DC trench, per metre
c_e^{ac}	1.0	AC trench, per metre
c_{ov}^{dc}	200.0	DC×AC overlap penalty
c_{ov}^{ac}	200.0	DC-trench / AC-internal overlap penalty
c_k^{inv}	per type	inverter install cost, read per inverter type

Table 5.1: Default cost coefficients used in the objective function across all experiments of this chapter. The inverter install cost c_k^{inv} is specified per inverter type and is reported together with the instance family that uses it.

The coefficients in Table 5.1 are not market prices. Cable, trenching, and overlap costs vary too widely across vendors and sites for any single figure to be representative; the values instead fix a deliberate ordering of magnitudes. Trenching dominates at 50 per metre, PV-string and AC cabling cost 1.0 per metre, DC homeruns are cheapest at 0.1, and the two overlap penalties are both 200. That ordering steers the solver toward layouts a site engineer would accept. Appendix A derives the relations among the coefficients.

5.2 Scaling on Synthetic Instances

This section answers the first research question of the chapter: how do the string-indexed and inverter-type-indexed formulations scale with the size of the plant, and which structural features of the instance dominate solver effort?

5.2.1 Synthetic Instance Family

Each experiment pairs a generated plant geometry with a set of inverter types. The generator produces only the geometry, fixed by two parameters: the number of arrays $N_A \in \mathbb{Z}_{\geq 1}$ and the number of transformers $N_T \in \{1, 2\}$. Sweeping N_A grows the plant; N_T fixes the structural variant under study.

The plant geometry follows a regular row-stack (Figure 5.1): each PV array hosts 24 modules along a 46-metre row, and successive rows sit 10 metres apart, so adding an array adds one row at a fixed horizontal extent. When $N_T = 1$ the transformer sits centred below row 0; when $N_T = 2$ the two transformers sit at the 25% and 75% horizontal positions and split the total module capacity. The remaining generator constants are fixed across all sizes; Appendix B lists them in full, together with the rationale behind the capacity bounds. Table 5.2 lists the available inverter types.

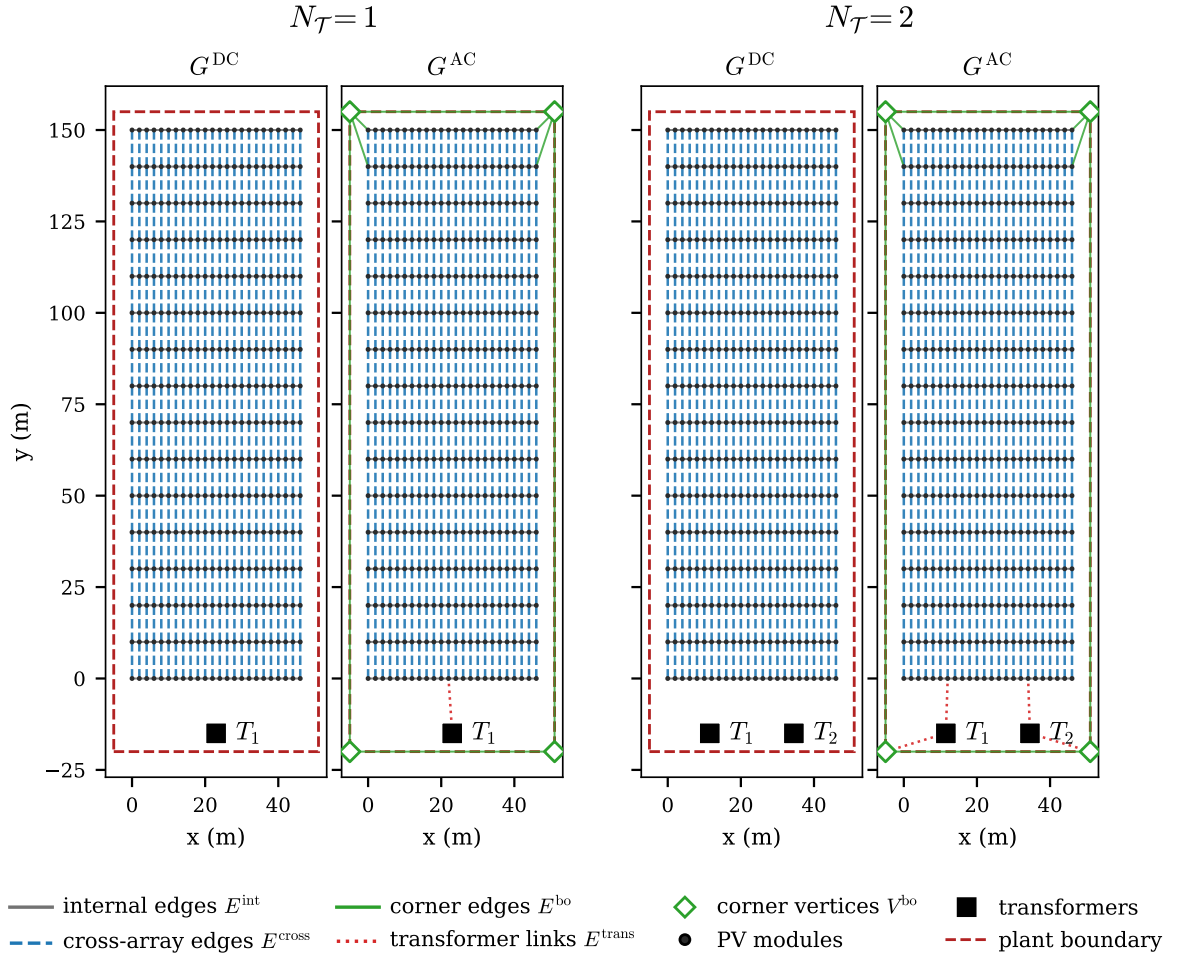


Figure 5.1: The synthetic instance at $N_A = 16$ under its two structural variants, each shown as the DC routing graph G^{DC} and the AC routing graph G^{AC} : $N_T = 1$ (left pair) and $N_T = 2$ (right pair). The 16 module rows stack vertically at a fixed horizontal extent; the transformers sit below row 0—centred when $N_T = 1$, at the 25% and 75% horizontal positions when $N_T = 2$.

$\mathcal{K}$	$S_k^{\min}$	$S_k^{\max}$	$Q_k^{\min}/Q_k^{\max}$	c_k^{inv}
small	1	1	24	10 000
medium	1	2	24	12 500
large	1	4	24	15 000

Table 5.2: Inverter types supplied to the solver. The per-type install cost c_k^{inv} scales sub-linearly in the maximum string count, so larger types carry a lower per-string cost.

5.2.2 Protocol

The study compares the string-indexed formulation of Chapter 3 and the inverter-type-indexed formulation of Chapter 4 across the three scenarios defined in Table 5.3. Scenario A serves as the baseline. Scenario B isolates the effect of inverter-type richness, and Scenario C shows the effect of adding a second transformer; all other factors are held fixed across the three.

Scenario	$N_{\mathcal{T}}$	$\mathcal{K}$	Purpose
A: Baseline	1	{small}	isolate the scaling effect of $N_{\mathcal{A}}$ under a single inverter type
B: Three-type catalogue	1	{small, medium, large}	evaluate both formulations under inverter type flexibility
C: Two transformers	2	{small}	isolate the effect of a second transformer on the AC-routing subproblem

Table 5.3: Scenarios for the scaling study. All three scenarios sweep $N_{\mathcal{A}} \in \{1, 4, 8, 16, 32, 64, 128\}$. The inverter types $\mathcal{K}$ refers to the entries defined in Table 5.2.

A *cell*: one of the two formulations, solved on one instance, where an instance is a single scenario at a single array count $N_{\mathcal{A}}$. The seven array counts and the two formulations therefore give 14 cells per scenario and 42 cells in total. Each cell is solved three times, once under each of the solver seeds 42, 1337, and 228. The wall-clock limit grows with the instance: 1200 seconds for $N_{\mathcal{A}} \leq 16$, then 1800, 2700, and 3600 seconds at $N_{\mathcal{A}} = 32, 64$, and 128 respectively. The string-indexed formulation is not run at $N_{\mathcal{A}} = 128$, since it already exhausts the memory of the test machine at $N_{\mathcal{A}} = 64$ (Section 5.2.3).

5.2.3 Model Size

Before any solver is invoked, the two formulations already differ markedly in the size of the program they generate (Figure 5.2). The string-indexed formulation couples a per-string variable to every routing arc, so its constraint count grows quadratically in $N_{\mathcal{A}}$ and identically across scenarios. The inverter-type-indexed formulation aggregates the strings over the type set $\mathcal{K}$, so its constraint count grows linearly, raised by roughly the factor $|\mathcal{K}|$ under the three-type catalogue of Scenario B. At $N_{\mathcal{A}} = 32$ the string-indexed model is around twelve times larger in the number of constraints than the inverter-type-indexed model and over 300 times larger in the number of non-zero entries in the constraint matrix, and at $N_{\mathcal{A}} = 64$ it reaches 386 million nonzeros and exhausts the 32 GB of the test machine in Scenarios B and C. Measured by nonzeros, the inverter-type-indexed model at $N_{\mathcal{A}} = 128$ is still around an order of magnitude smaller than the string-indexed model at $N_{\mathcal{A}} = 16$. Appendix C breaks these totals down by constraint family: each family grows with the index set it is generated over, which confirms that the quadratic growth is a property of the formulation, not of the instance graph.

5.2.4 LP Relaxation

Model size bounds the memory a formulation needs; how tight its linear relaxation is decides how much branch-and-bound search a solve has to do. To isolate relaxation strength from the integer search, the root linear relaxation of each formulation is solved on its own, under settings identical for both and a generous 1800-second limit (Table 5.4). The string-indexed relaxation gives the tighter bound on every instance, though only modestly: by 0.45% in the single-inverter-type scenarios and by 1.20% under the three inverter types, where the per-inverter-type aggregation loosens the relaxation most. That advantage is unusable at scale: the string-indexed LP alone exhausts the limit from $N_{\mathcal{A}} = 32$ on, whereas the inverter-type-indexed LP solves within seconds through $N_{\mathcal{A}} = 128$.

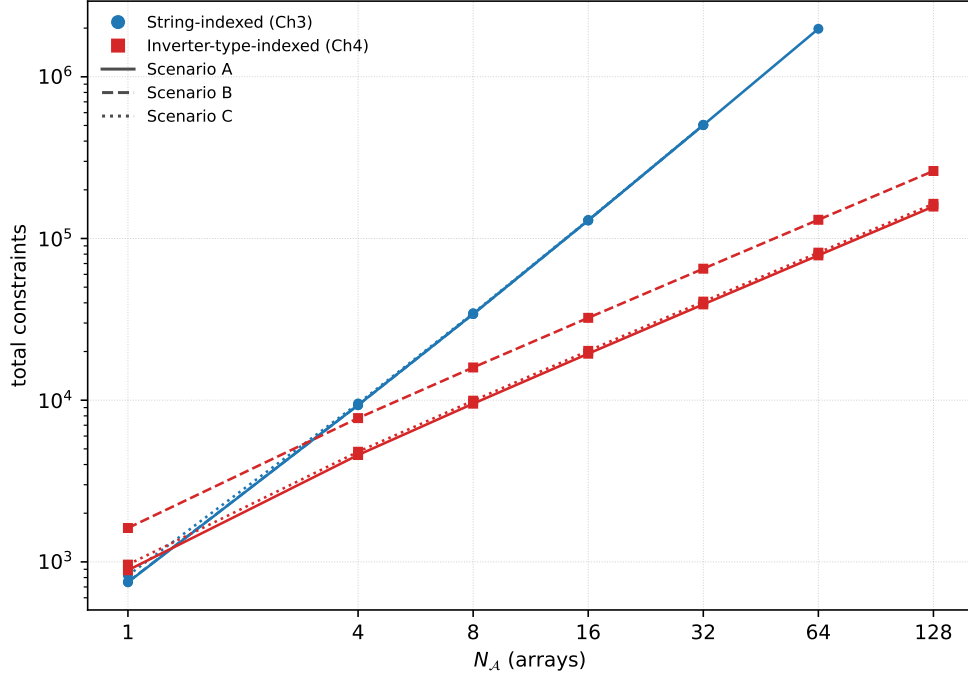


Figure 5.2: Total constraint count of both formulations against instance size (log-log); formulation is shown by colour and marker, scenario by line style. The string-indexed count is scenario-invariant and grows quadratically; the inverter-type-indexed count grows linearly, with the three-type catalogue of Scenario B sitting highest. Per-cell counts are listed in Table C.2.

Scn	N_A	String-indexed (Ch3)		Inverter-type-indexed (Ch4)		Δ (%)
		LP bound	t (s)	LP bound	t (s)	
A	1	10 051	0.01	10 006	0.01	0.45
A	4	40 219	0.51	40 040	0.05	0.45
A	8	80 479	24.15	80 119	0.07	0.45
A	16	161 118	354	160 399	0.12	0.45
A	32	<i>t/o</i>	1804	321 437	0.29	—
A	64	—	—	645 434	0.71	—
A	128	—	—	1 301 108	1.45	—
B	1	10 051	0.01	10 006	0.02	0.45
B	4	15 193	0.57	15 011	0.10	1.20
B	8	30 397	9.37	30 031	0.22	1.20
B	16	60 833	452	60 102	0.73	1.20
B	32	<i>t/o</i>	1812	120 364	1.57	—
B	64	—	—	241 368	4.07	—
B	128	—	—	485 296	82.37	—
C	1	<i>inf</i>	0.01	<i>inf</i>	0.01	—
C	4	40 216	0.43	40 035	0.07	0.45
C	8	80 471	6.92	80 110	0.09	0.45
C	16	161 102	74.54	160 379	0.21	0.45
C	32	<i>t/o</i>	1805	321 398	0.47	—
C	64	—	—	645 357	1.10	—
C	128	—	—	1 300 954	3.01	—

Table 5.4: LP-relaxation metrics under a controlled setting identical for both formulations (Gurobi 13.0.2, automatic LP method, 1800s limit). *LP bound* is the objective of the root linear relaxation—a lower bound on the integer optimum—and t its solve time; $\Delta = (b_{\text{Ch3}} - b_{\text{Ch4}})/b_{\text{Ch3}}$ is the relative looseness of the inverter-type-indexed relaxation. The string-indexed relaxation is consistently tighter (positive Δ), but its LP solve time grows steeply and exhausts the limit beyond $N_A = 16$; the inverter-type-indexed LP stays cheap through $N_A = 128$. *t/o*: solve hit the time limit; *inf*: relaxation infeasible (a single array cannot feed two transformers).

5.2.5 Scaling Results

Solve time separates the formulations only on the smallest instances: from $N_A = 16$ on, every cell exhausts its time budget. The comparison therefore rests on the optimality gap at termination, plotted in Figure 5.3; the per-cell outcomes are listed in Table C.2.

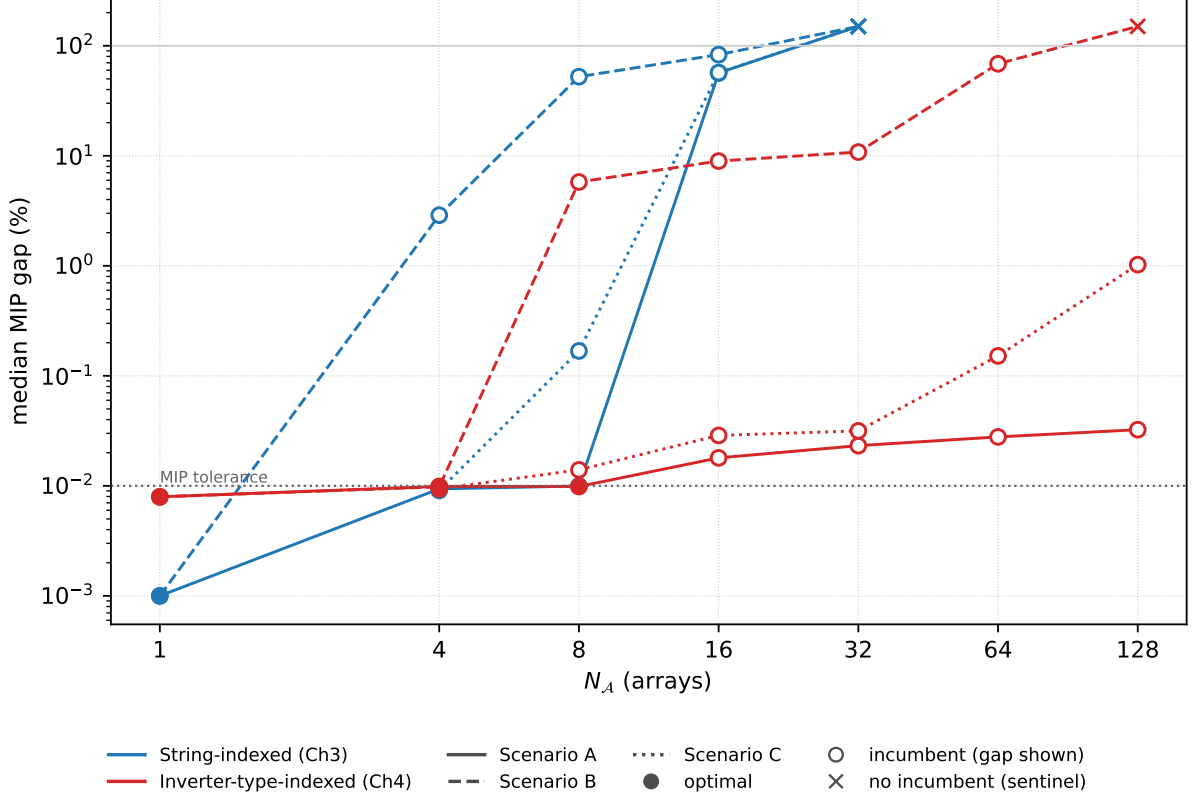


Figure 5.3: Median optimality gap against instance size, all three scenarios on one axes (log-log). Formulation is shown by colour and scenario by line style; the marker gives each cell’s outcome—filled for an optimal solve, hollow for a time-limited incumbent (gap shown), and a cross at a sentinel above 100% for a cell with no incumbent. The dotted line marks the 10^{-4} MIP tolerance. The string-indexed formulation loses its incumbent by $N_A = 32$ in every scenario, whereas the inverter-type-indexed formulation holds a near-optimal gap to far larger instances in the single-type Scenarios A and C.

Scenario A. Both formulations solve $N_A \leq 8$ to optimality or within a fraction of a percent. Beyond that they part. The inverter-type-indexed gap stays below 0.04% all the way to $N_A = 128$, while the string-indexed incumbent collapses at $N_A = 16$ to a 56% gap—its best solution costs 2.3 times the optimum the other formulation reaches—and is absent from $N_A = 32$ on.

Scenario B. The inverter three-type is the hardest setting for both formulations: at every size the gaps sit orders of magnitude above Scenario A, so inverter type richness, not plant size, is the dominant source of difficulty. The string-indexed formulation already times out at $N_A = 4$ (a 3.2% gap) and degrades to 83% by $N_A = 16$; the inverter-type-indexed formulation still solves $N_A = 4$ to optimality and holds a 6–11% gap through $N_A = 32$, before its incumbent also degrades (69% at $N_A = 64$) and disappears at $N_A = 128$.

Scenario C. The single-array instance is infeasible, as two transformers cannot share one array’s single string. At $N_A = 4$ both formulations reach optimality, and the string-indexed one is the faster (32 against 203 seconds); it is the last size at which it leads. From $N_A = 8$ the string-indexed formulation repeats its Scenario A trajectory—a 57% gap at $N_A = 16$, no incumbent at $N_A = 32$, out of memory at $N_A = 64$ —whereas the second transformer costs the inverter-type-indexed formulation little: its gap stays below 0.04% through $N_A = 32$, as in Scenario A, and widens only on the largest plants, to 0.2% at $N_A = 64$ and 1.8% at $N_A = 128$.

Across the three scenarios the picture is consistent. The string-indexed formulation is competitive only while its quadratic size stays small: past $N_A = 8$ its incumbents are far from optimal or absent, and at $N_A = 64$ the model no longer fits in memory. The inverter-type-indexed formulation sustains near-optimal

incumbents on instances an order of magnitude larger, and within the time budget it is defeated only by inverter-type richness, not by plant size.

5.3 Where the Monolithic MILP Struggle

This section answers the second research question of the chapter: on real plant layouts, which subset of the decision variables is the tractability bottleneck for the monolithic MILP? The scaling study of Section 5.2 varied plant size on synthetic geometry; here the geometry is fixed—each instance is a real operating plant—and the experiment instead varies *which decisions the solver must still make*. That study also settles which formulation to test here: the string-indexed model of Chapter 3 loses its incumbent by $N_A = 32$ arrays (768 modules) in every scenario and exhausts the machine’s memory at $N_A = 64$ (Section 5.2.5). The real plants are more complex: 8 out of 9 exceed that 768-module mark. All freezing experiments therefore use the inverter-type-indexed formulation of Chapter 4.

5.3.1 Instances and Reference Annotations

The instances are nine operating utility-scale plants supplied by Spanish company as engineering layout drawings. Each drawing was manually transcribed into the site model of Chapter 2—module rows, the boundary polygon, ground obstacles, transformer positions, and the available inverter types—from which the DC and AC graphs of Chapter 3 are computed. Table 5.5 reports the properties of the nine instances. Instances are labelled P1–P9 in order of module count.

Inst.	Site model				Routing graphs		Annotation	
	N_M	N_A	N_T	N_K	$ \vec{E}^{DC} $	$ \vec{E}^{AC} $	Str.	Inv.
P1	313	10	1	1	976	1160	18	3
P2	1180	32	1	2	5624	6348	46	5
P3	1296	14	1	1	5464	5710	50	5
P4	1317	18	1	1	5316	5702	54	5
P5	2224	24	1	1	9572	9660	90	9
P6	2664	32	1	1	12344	12548	110	11
P7	3357	143	2	3	14652	14554	134	14
P8	3517	62	2	3	14122	14236	141	14
P9	4889	41	2	3	18884	19052	202	18

Table 5.5: Real-world instance set, ordered by module count. The site columns use the notation of Section 2.3: N_M modules in N_A arrays, N_T transformer locations, and a catalogue of N_K inverter types. $|\vec{E}^{DC}|$ and $|\vec{E}^{AC}|$ are the arc counts of the directed DC and AC graphs, which set the size of the edge-indexed routing variables. The annotation columns are not layout inputs but outcomes of the reference annotation: *Str.* counts its non-empty PV-strings R_j and *Inv.* its inverter placements $\mathcal{I}$ (Section 2.5); they are reproduced here so that the freeze targets can be read against the instance. The resulting model dimensions are listed in Appendix D.1, and each layout is plotted in Appendix E.

The set spans an order of magnitude in module count, and one instance stands apart from that ordering: P7’s modules are split across 143 short, geographically separated arrays, which inflates the cable-crossing conflict set to 6 058 cross-cross pairs against at most 451 on any other instance—the structural reason it is the hardest cell.

For each plant, an engineer-style reference solution was annotated by hand: the modules were partitioned into PV-strings, each string was assigned its two terminals, an inverter was placed for each group of strings, and every string was assigned to one of those inverters. The annotation thus fixes three decision classes exactly—PV-string formation (with its terminals), inverter placement, string-to-inverter assignment.

5.3.2 Freeze Hierarchy and Protocol

A freeze configuration fixes a subset of the annotated decision variables to their reference values—setting each variable’s bounds to its annotated value before the solve—and leaves the rest free. Inverter placement and PV-string formation are independent, which gives the four configurations of Table 5.6: the baseline F_0 , the single freezes F_1 (placement) and F_2 (PV-strings), and their union F_3 .

Config.	Frozen decisions	Variables the solver still chooses
F_0	none (baseline)	placement, strings, assignment, all routing
F_1	inverter placement	strings, assignment, all routing
F_2	PV-strings (+ terminals)	placement, assignment, all routing
F_3	PV-strings + inverter placement	assignment, all routing

Table 5.6: Freeze configurations, from the full model (F_0) to the most constrained (F_3). Each fixes the listed decisions to their reference-annotation values and optimises the rest; routing (DC-homeruns, AC-feeders) is never frozen.

The solver configuration and cost coefficients are those of Section 5.1. Every solve is given 3600 seconds of wall clock. For each cell three quantities are recorded: solve time (Gurobi `Runtime`), final optimality gap, and best incumbent. Per-cell numbers are tabulated in Appendix D.2.

5.3.3 Results

Figure 5.4 summarises all 36 cells. Three findings emerge.

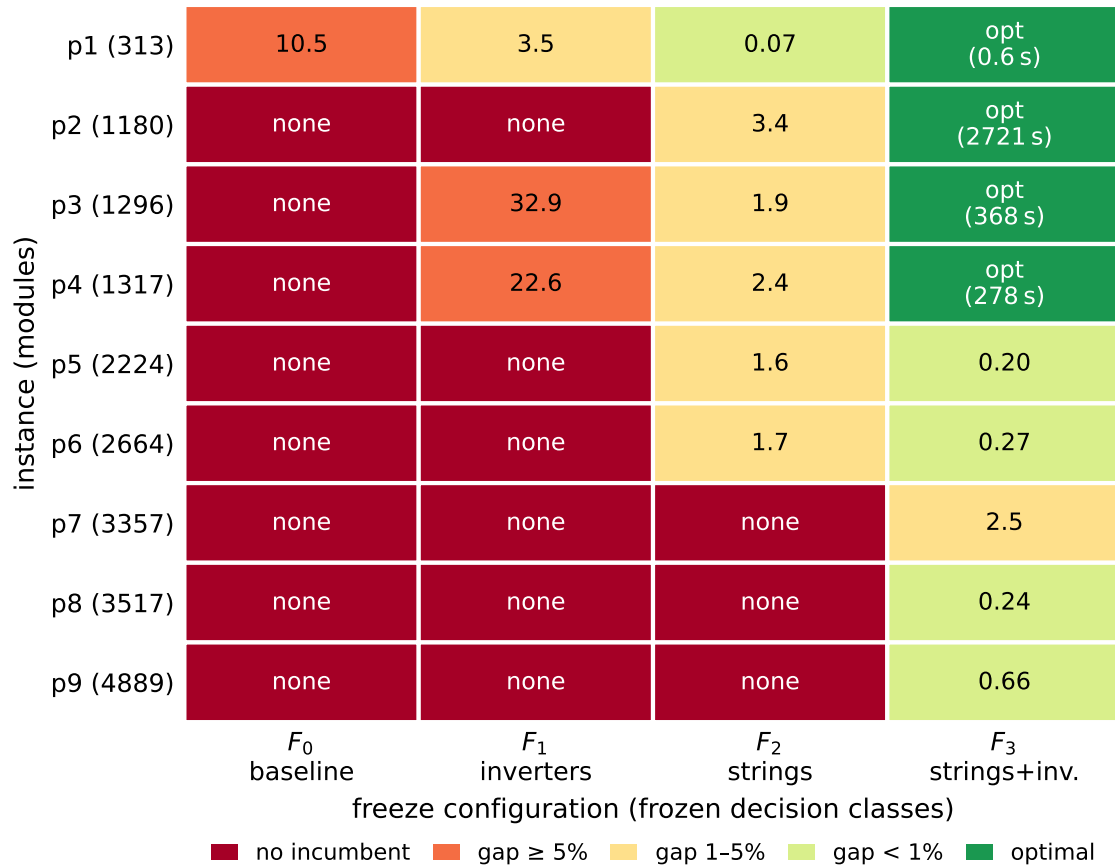


Figure 5.4: Outcome of every (instance, freeze configuration) cell. Each cell shows the final MIP gap in percent, *opt* with the solve time when the solver certified optimality, or *none* when the time limit expired without any incumbent. Colour encodes the outcome class. Instances are ordered by module count; configurations by freeze depth as in Table 5.6.

Feasibility, not optimality, is the failure mode. The baseline F_0 terminates without *any* feasible solution on eight of the nine plants: after an hour, the solver has not produced a single layout, let alone a certificate. Only the smallest instance P1 (313 modules) yields an incumbent, and still carries a 10.5% gap at timeout. The monolithic model does not degrade gracefully with instance size; past a few hundred modules it returns no incumbent at all.

Strings are the bottleneck, not inverter placement. Freezing inverter placement alone (F_1) barely helps: it yields an incumbent on only three plants—P1 (3.5%), P3 (32.9%), and P4 (22.6%)—and the two larger of these still carry gaps above 20%. Freezing PV-string formation alone (F_2) instead restores feasibility on the six smaller plants, up to 2664 modules, with gaps below 3.5%. The relaxation explains why: where the root gap is measurable it drops from 21–36% under F_0 and F_1 to 0.1–4.4% once the string variables are fixed (Appendix D.2). The string-formation variables—membership, terminals, and the trail arcs that connect modules into strings—are what keeps the relaxation weak.

Placement and partitioning are coupled. On the three largest plants neither single freeze produces a solution, yet freezing both (F_3) yields incumbents on all nine instances, with gaps below 0.7% everywhere except the fragmented P7 (2.5%, the largest residual gap) and optimality certificates on P1, P2, P3, and P4. The first incumbent arrives within 13 seconds on eight of the nine plants—P7 included, despite its conflict structure—and only P9 takes longer, at 52 seconds.

Together the three findings rank the decision classes: PV-string formation dominates the difficulty, inverter placement matters mainly through its coupling with the strings, and the residual problem once both are fixed—assignment plus routing—closes to sub-percent gaps within minutes.

Chapter 6

Staged Decomposition Solver

Chapter 4 couples two decision layers in a single MILP: a *topology* layer chooses inverter sites, DC trenches, and AC feeder routes, and a *string* layer fills in PV-string composition and DC homerun circuits under that topology. The string layer depends on the topology, yet the monolithic model branches on both.

This chapter decomposes the problem into four stages, each handing its solution to the next as a partial variable fixing or a warm-start hint. Stage 1 fixes the topology in an aggregated single-commodity MILP; Stage 2 pre-routes each non-inverter module to its assigned inverter by a multi-commodity flow on the Stage 1 graph; Stage 3 reinstates Chapter 4 with the topology frozen and the assignment warm-started; Stage 4 lifts the freeze and re-solves from the Stage 3 incumbent. Sections 6.1–6.4 develop the stages; Section 6.5 summarises the staged decomposition; Section 6.6 evaluates it against the complete human annotation on the nine operating plants.

6.1 Stage 1: Aggregated Flow MILP

Stage 1 fixes the topology (inverter placements, DC trenches, transformer assignments, and AC feeders) and routes a single-commodity DC flow over it.

Stage 1 constraints carry $F\#$ tags, distinct from Chapter 4’s $C\#$.

Inverter Placement

Stage 1 inherits the inverter-placement variables $y_{k,m}$ and u_k together with Constraints (C1), (C2.1), (C2.2), and (C2.3) of Chapter 4, unchanged.

Inverter Capacity

Chapter 4 bounds the per-type module load $L_{m,k}$ through Constraint (C10.2). Stage 1 lacks the per-type breakdown and applies the same envelope to the absorbed module count.

$\text{load}_m \in \mathbb{Z}_{\geq 0}$ Number of non-inverter modules absorbed by the inverter at location m ,
 $\forall m \in \mathcal{M}$.

Module Absorption Envelope. The inverter’s own location accounts for one of them and does not source flow, so the count of *absorbed* modules lies in $[S_k^{\min} Q_k^{\min} - 1, S_k^{\max} Q_k^{\max} - 1]$. At non-inverter locations the placement indicator $\sum_k y_{k,m}$ vanishes and both bounds collapse to $\text{load}_m = 0$.

$$\sum_{k \in \mathcal{K}} (S_k^{\min} Q_k^{\min} - 1) y_{k,m} \leq \text{load}_m \leq \sum_{k \in \mathcal{K}} (S_k^{\max} Q_k^{\max} - 1) y_{k,m} \quad \forall m \in \mathcal{M} \quad (\text{F1})$$

DC Flow Conservation

Stage 1 routes the DC current as a single commodity on $\vec{G}^{\text{DC}}$, in place of the per-string, per-type flow $f_{e,k}$ of Constraint (C4.1). The aggregated flow is sourced at modules, absorbed at inverters, and routed by flow conservation without distinguishing which strings or homeruns it represents.

$f_e \in \mathbb{R}_{\geq 0}$ Single-commodity DC flow on arc e ,
 $\forall e \in \vec{E}^{\text{DC}}$.

Aggregated Flow Conservation. Each non-inverter module is a source of one unit; each inverter at m is a sink of load_m units. The placement indicator $\sum_k y_{k,m}$ encodes both cases in a single linear equation.

$$\sum_{e \in E_{\text{DC}}^+(m)} f_e - \sum_{e \in E_{\text{DC}}^-(m)} f_e = 1 - \sum_{k \in \mathcal{K}} y_{k,m} - \text{load}_m \quad \forall m \in \mathcal{M} \quad (\text{F2})$$

DC Arc Utilisation

The aggregated flow f_e could traverse a DC edge in both directions at once. To admit at most one direction per edge, Stage 1 introduces a binary arc-usage indicator that any positive flow activates.

$\chi_e \in \{0, 1\}$ Equals 1 if arc e carries positive DC flow,
 $\forall e \in \vec{E}^{\text{DC}}$.

Arc Utilisation Activation. The Big-M $N_{\mathcal{M}}$ bounds f_e , since at most $N_{\mathcal{M}}$ units of flow are sourced; any positive flow then forces $\chi_e = 1$.

$$f_e \leq N_{\mathcal{M}} \chi_e \quad \forall e \in \vec{E}^{\text{DC}} \quad (\text{F3.1})$$

Arc Direction Consistency. At most one arc direction of any undirected DC edge carries flow. Together with the activation linkage, this rules out flow in both directions.

$$\chi_{(u,v)} + \chi_{(v,u)} \leq 1 \quad \forall \{u, v\} \in E^{\text{DC}} \quad (\text{F3.2})$$

DC Trench Activation

Chapter 4's Constraints (C13.1)–(C13.3) model DC trenching through an integer trench count τ_e^{DC} and a binary utilisation indicator η_e . The aggregated flow of Stage 1 counts modules rather than cables, so the trench count cannot be recovered. Stage 1 retains only the binary η_e , which now records whether a trench is installed on the edge.

$\eta_e \in \{0, 1\}$ Equals 1 if a DC trench is installed on edge e ,
 $\forall e \in E_{\text{prune}}^{\text{cross}}$.

Edge Activation. Constraints (C13.1)–(C13.2) reduce to a single Big-M activation linkage on the bidirectional aggregated flow. At most $N_{\mathcal{M}}$ units of flow are sourced.

$$f_{(u,v)} + f_{(v,u)} \leq N_{\mathcal{M}} \eta_e \quad \forall e = \{u, v\} \in E_{\text{prune}}^{\text{cross}} \quad (\text{F4.1})$$

No DC Crossings. Constraint (C13.3) applies unchanged.

$$\eta_{e_1} + \eta_{e_2} \leq 1 \quad \forall \{e_1, e_2\} \in \mathcal{O}^{\text{cross} \times \text{cross}} \quad (\text{F4.2})$$

AC Routing

Stage 1 inherits Constraints (C9.1) (transformer assignment) and (C12.1)–(C12.5) (AC flow conservation, no transformer transit, edge activation, direction consistency, and trench capacity) of Chapter 4 unchanged. The per-transformer flow $\alpha_{e,t}$ is relaxed from integer to continuous; the trench count τ_e^{AC} and the directed arc-utilisation indicator α_e^{AC} retain their integer and binary domains.

AC/DC Overlap Penalties

Stage 1 inherits Constraints (C14) (AC/DC overlap) and (C15) (DC-trench / AC-internal overlap) of Chapter 4 unchanged.

Objective Function

The Stage 1 objective minimises six cost terms: DC trenches, AC trenches, DC cabling, inverter cost, and the two geometric-overlap penalties. All but the cabling term reuse Chapter 4's coefficients; the cabling term replaces the per-string and per-homerun charges of Constraint (OBJ) by a single charge on the aggregated arc flow f_e . This proxy over-counts arcs shared by several inverters; Stage 2 recovers the exact per-cable cost.

$$\begin{aligned}
 \min Z_1 = & \underbrace{\sum_{e \in E_{\text{prune}}^{\text{cross}}} c_e^{\text{trench}} w_e \eta_e}_{\text{DC-trench cost}} + \underbrace{\sum_{e \in E^{\text{AC}}} c_e^{\text{AC}} w_e \tau_e^{\text{AC}}}_{\text{AC-trench cost}} \\
 & + \underbrace{\sum_{e \in \vec{E}^{\text{DC}}} c_e^{\text{DV}} w_e f_e}_{\text{DC-cable cost}} + \underbrace{\sum_{k \in \mathcal{K}} \sum_{m \in \mathcal{M}} c_k^{\text{inv}} y_{k,m}}_{\text{Inverter cost}} \\
 & + \underbrace{\sum_{(e^{\text{DC}}, e^{\text{AC}}) \in \mathcal{O}^{\text{cross} \times \text{AC}}} c_{(e^{\text{DC}}, e^{\text{AC}})}^{\text{ov}} o_{e^{\text{DC}}, e^{\text{AC}}}}_{\text{AC/DC overlap penalty}} + \underbrace{\sum_{\omega \in \mathcal{O}^{\text{cross} \times \text{int}}} c_{\omega}^{\text{tov}} o_{\omega}}_{\text{Trench/AC-internal overlap penalty}} \quad (\text{OBJ-S1})
 \end{aligned}$$

6.2 Stage 2: Path Resolution MILP

Stage 1's aggregated DC flow f_e counts the modules traversing arc e but does not record which inverter absorbs each unit.

Given Stage 1's placement y , Stage 2 solves a multi-commodity DC flow on the DC graph: each non-inverter module is assigned to one inverter, and per-inverter flow conservation pins the arcs that carry its flow. Let $M^{\text{inv}} = \{m \in \mathcal{M} \mid \sum_{k \in \mathcal{K}} y_{k,m} = 1\}$ denote the active inverter locations, and for each $m' \in M^{\text{inv}}$ let $k(m') \in \mathcal{K}$ be the type placed there, that is, the unique k with $y_{k,m'} = 1$. Routing is carried out on the full directed DC arc set $\vec{E}^{\text{DC}}$ —both orientations of every internal and candidate cross edge—so that Stage 2 may reopen trenches Stage 1 left closed.

Stage 2 constraints are tagged $P\#$.

Module Assignment

Stage 2 pins every non-inverter module to a single inverter and exposes the resulting per-inverter module count as an explicit variable.

$$\begin{aligned}
 s_{m,m'} & \in \{0, 1\} && \text{Equals 1 if non-inverter module } m \text{ is served by the inverter at } m', \\
 & && \forall m \in \mathcal{M} \setminus M^{\text{inv}}, \forall m' \in M^{\text{inv}}. \\
 \text{load}_{m'} & \in \mathbb{Z}_{\geq 0} && \text{Number of non-inverter modules served by the inverter at } m', \\
 & && \forall m' \in M^{\text{inv}}.
 \end{aligned}$$

Assignment Partition. Each non-inverter module is served by exactly one inverter.

$$\sum_{m' \in M^{\text{inv}}} s_{m,m'} = 1 \quad \forall m \in \mathcal{M} \setminus M^{\text{inv}} \quad (\text{P1.1})$$

Load Coupling. The load on each inverter equals the count of modules assigned to it.

$$\sum_{m \in \mathcal{M} \setminus M^{\text{inv}}} s_{m,m'} = \text{load}_{m'} \quad \forall m' \in M^{\text{inv}} \quad (\text{P1.2})$$

Inverter Capacity

With the inverter type $k(m')$ now pinned at every active location, the Stage 1 capacity envelope of Constraint (F1) collapses from a per-type sum to a scalar bound:

$$S_{k(m')}^{\min} Q_{k(m')}^{\min} - 1 \leq \text{load}_{m'} \leq S_{k(m')}^{\max} Q_{k(m')}^{\max} - 1 \quad \forall m' \in M^{\text{inv}} \quad (\text{P2.1})$$

Per-Destination Flow Conservation

One commodity per inverter routes the modules assigned to that inverter. The aggregated flow f_e of Constraint (F3.1) is recovered as the sum of these commodities over $m' \in M^{\text{inv}}$.

$$g_{e,m'} \in \mathbb{R}_{\geq 0} \quad \text{Flow on arc } e \text{ destined for the inverter at } m', \\ \forall e \in \vec{E}^{\text{DC}}, \forall m' \in M^{\text{inv}}.$$

Per-Destination Flow Conservation. A non-inverter module n sources one unit of commodity m' when assigned to that inverter and zero otherwise; the destination inverter absorbs $\text{load}_{m'}$ units; every other inverter transmits the commodity unchanged.

$$\sum_{e \in E_{\text{DC}}^+(n)} g_{e,m'} - \sum_{e \in E_{\text{DC}}^-(n)} g_{e,m'} = \begin{cases} s_{n,m'} & \text{if } n \in \mathcal{M} \setminus M^{\text{inv}}, \\ -\text{load}_{m'} & \text{if } n = m', \\ 0 & \text{if } n \in M^{\text{inv}} \setminus \{m'\}, \end{cases} \quad \forall n \in \mathcal{M}, m' \in M^{\text{inv}} \quad (\text{P3.1})$$

Per-Destination Arc Utilisation

Stage 2 introduces one binary per (arc, inverter) pair, so that the objective charges exactly one cable for each pair in use.

$$\chi_{e,m'} \in \{0, 1\} \quad \text{Equals 1 if arc } e \text{ carries positive flow of commodity } m', \\ \forall e \in \vec{E}^{\text{DC}}, \forall m' \in M^{\text{inv}}.$$

Arc Utilisation Activation. An inverter of type k absorbs at most $S_k^{\max} Q_k^{\max} - 1$ units of its own commodity, so the Big-M

$$g^{\max} = \max_{m' \in M^{\text{inv}}} (S_{k(m')}^{\max} Q_{k(m')}^{\max} - 1)$$

bounds every flow $g_{e,m'}$ and forces $\chi_{e,m'} = 1$ whenever $g_{e,m'} > 0$. The reverse implication follows from the objective's positive coefficient on $\chi_{e,m'}$.

$$g_{e,m'} \leq g^{\max} \chi_{e,m'} \quad \forall e \in \vec{E}^{\text{DC}}, m' \in M^{\text{inv}} \quad (\text{P4.1})$$

DC Trench Activation

Stage 2 promotes the same indicator η_e from Constraint (F4.1) to a decision variable, supplied as a MIP start from the Stage 1 solution. Per-inverter flow now reveals which cross-array edges need a trench.

$$\eta_e \in \{0, 1\} \quad \text{Equals 1 if a DC trench is installed on edge } e, \\ \forall e \in E_{\text{prune}}^{\text{cross}}.$$

Edge Activation. A cross-array edge can carry flow only if its trench is installed. Summing $g_{e,m'}$ over both orientations and all destination inverters gives the total cable count on the edge, at most $N_{\mathcal{M}}$; this bound serves as the Big-M.

$$\sum_{m' \in M^{\text{inv}}} (g_{(u,v),m'} + g_{(v,u),m'}) \leq N_{\mathcal{M}} \eta_e \quad \forall e = \{u, v\} \in E_{\text{prune}}^{\text{cross}} \quad (\text{P5.1})$$

No DC Crossings. The geometric incompatibility carried over from Constraint (F4.2) forbids installing two crossing trenches simultaneously.

$$\eta_{e_1} + \eta_{e_2} \leq 1 \quad \forall \{e_1, e_2\} \in \mathcal{O}^{\text{cross} \times \text{cross}} \quad (\text{P5.2})$$

AC/DC Overlap Penalties

Reopening the DC trenches reactivates the geometric coupling between DC and AC cabling, so Stage 2 reinstates Constraints (C14) and (C15) of Chapter 4. With the AC routing frozen at its Stage 1 value $\bar{\alpha}^{\text{AC}}$, each penalty depends only on the reopened trench indicators of Constraint (P5.1), and the positive objective coefficients c^{ov} and c^{tov} keep each indicator at 0 unless its forcing inequality lifts it.

$$\begin{aligned} o_{e^{\text{DC}}, e^{\text{AC}}} &\in \{0, 1\} && \text{Equals 1 if the DC trench } e^{\text{DC}} \text{ overlaps the active AC edge } e^{\text{AC}}, \\ &&& \forall (e^{\text{DC}}, e^{\text{AC}}) \in \mathcal{O}^{\text{cross} \times \text{AC}}. \\ o_{\omega} &\in \{0, 1\} && \text{Equals 1 if the DC trench and AC cables overlap at the module identified by 4-tuple } \\ &&& \omega, \\ &&& \forall \omega \in \mathcal{O}^{\text{cross} \times \text{int}}. \end{aligned}$$

AC/DC Overlap. The indicator is forced to 1 when the trench e^{DC} is installed and the frozen AC edge $e^{\text{AC}} = \{u, v\}$ carries flow in either direction. The frozen sum $\bar{\alpha}_{(u,v)}^{\text{AC}} + \bar{\alpha}_{(v,u)}^{\text{AC}} \in \{0, 1\}$ acts as the undirected-edge utilisation indicator.

$$o_{e^{\text{DC}}, e^{\text{AC}}} \geq \eta_{e^{\text{DC}}} + \bar{\alpha}_{(u,v)}^{\text{AC}} + \bar{\alpha}_{(v,u)}^{\text{AC}} - 1 \quad \forall (e^{\text{DC}}, e^{\text{AC}} = \{u, v\}) \in \mathcal{O}^{\text{cross} \times \text{AC}} \quad (\text{P6.1})$$

DC-Trench / AC-Internal Overlap. For each 4-tuple $\omega = (e^+, e^-, e_1^{\text{int}} = \{u_1, v_1\}, e_2^{\text{int}} = \{u_2, v_2\})$, the indicator is forced to 1 when both cross-array edges carry trenches and both internal edges carry frozen AC flow.

$$o_{\omega} \geq \eta_{e^+} + \eta_{e^-} + \bar{\alpha}_{(u_1, v_1)}^{\text{AC}} + \bar{\alpha}_{(v_1, u_1)}^{\text{AC}} + \bar{\alpha}_{(u_2, v_2)}^{\text{AC}} + \bar{\alpha}_{(v_2, u_2)}^{\text{AC}} - 3 \quad \forall \omega \in \mathcal{O}^{\text{cross} \times \text{int}} \quad (\text{P6.2})$$

Objective Function

Stage 2 minimises the DC-cable cost—now charged per used (arc, inverter) pair through $\chi_{e,m'}$ —together with the trench cost and the two overlap penalties of Constraints (P6.1) and (P6.2). The inverter cost stays at its Stage 1 value, to be reconciled in Stage 4 once the full topology reopens.

$$\begin{aligned} \min Z_2 = & \underbrace{\sum_{e \in \vec{E}^{\text{DC}}} \sum_{m' \in M^{\text{inv}}} c_e^{\text{pv}} w_e \chi_{e,m'}}_{\text{DC-cable cost}} + \underbrace{\sum_{e \in E_{\text{prune}}^{\text{cross}}} c_e^{\text{trench}} w_e \eta_e}_{\text{DC-trench cost}} \\ & + \underbrace{\sum_{(e^{\text{DC}}, e^{\text{AC}}) \in \mathcal{O}^{\text{cross} \times \text{AC}}} c^{\text{ov}} o_{e^{\text{DC}}, e^{\text{AC}}}}_{\text{AC/DC overlap}} + \underbrace{\sum_{\omega \in \mathcal{O}^{\text{cross} \times \text{int}}} c^{\text{tov}} o_{\omega}}_{\text{trench / AC-internal overlap}} \end{aligned} \quad (\text{OBJ-S2})$$

6.3 Stage 3: Topology-Frozen Chapter 4 MILP

Stage 3 reinstates the inverter-type-indexed MILP of Chapter 4 with four topology decisions frozen: the placement $y_{k,m}$, type activation u_k , and inverter-to-transformer assignment $v_{m,t}$ are clamped to the Stage 1 solution, and the DC trench indicator η_e to the Stage 2 solution, in each case by tightening the variable's lower and upper bounds to that value. Every other variable is re-solved within the frozen topology, under the complete Chapter 4 constraint set.

The AC routing $(\tau_e^{\text{AC}}, \alpha_e^{\text{AC}})$ is hinted rather than frozen: Stage 1 relaxes $\alpha_{e,t}$ to continuous, and an integer-exact freeze of α_e^{AC} to a fractional total would be infeasible under Constraint (C12.4). The PV-string variables $s_{m,k}$ and $f_{e,k}$, together with the per-inverter load $L_{m,k}$, are hinted from Stage 2's assignment $s_{m,m'}$ and arc-usage indicators $\chi_{e,m'}$.

6.4 Stage 4: Polished Chapter 4 MILP

Stage 4 re-solves the Chapter 4 MILP with no frozen variables. Stage 3’s incumbent is supplied as a MIP-start seed on every variable. With the topology unlocked, the solver can revise the placement and trench decisions that Stages 1 and 2 committed before the string-level costs were visible.

6.5 Decomposition Summary

The four stages execute in sequence. Stage 1 produces a topology; Stage 2 assigns each module to an inverter and revises the trench layout under per-inverter flow; Stage 3 reinstates the Chapter 4 MILP with the placement frozen at Stage 1, the trenches at Stage 2, and the assignment supplied as a warm start; Stage 4 lifts the freeze and re-solves the full Chapter 4 MILP from the Stage 3 incumbent.

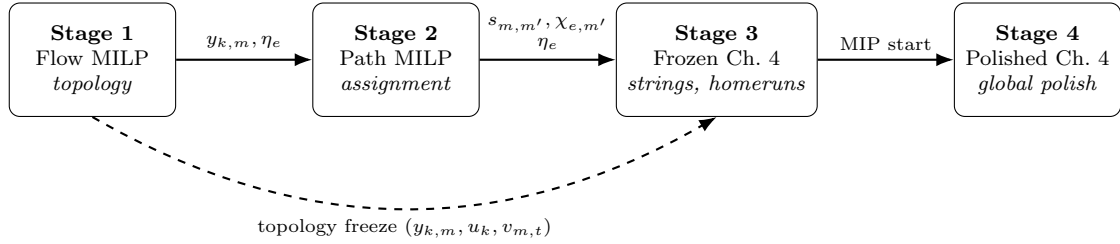


Figure 6.1: Four-stage decomposition. Solid arrows carry the variable handover in the linear $S_1 \rightarrow S_2 \rightarrow S_3 \rightarrow S_4$ flow. The dashed arrow marks the topology freeze: Stage 1’s $y_{k,m}$, u_k , and $v_{m,t}$ are clamped at Stage 3 directly—of the three, only $y_{k,m}$ also enters Stage 2, as fixed input. The trench indicator η_e is reopened in Stage 2 and clamped at Stage 3 from there. The AC routing (τ_e^{AC} , α_e^{AC}) is omitted from the diagram: Stage 2 consumes the frozen α_e^{AC} in its overlap penalties, and Stage 3 receives both variables as hints. The full handover is detailed in Table 6.1.

The handover is detailed in Table 6.1. “Frozen” marks a value from the preceding stage clamped by tightening the variable’s bounds to that single value. “Hinted” marks a value supplied as a MIP-start seed and left free for the solver to revise. “Decided” marks variables that the stage solves from scratch. “Input” marks a value consumed as fixed data rather than as a decision variable, and “—” marks a variable group absent from the stage’s model.

Variable group	Stage 2	Stage 3	Stage 4
Topology $y_{k,m}, u_k, v_{m,t}$	input ($y_{k,m}$ only)	frozen (S1)	hinted
DC trench η_e	hinted (S1)	frozen	hinted
AC routing $\tau_e^{\text{AC}}, \alpha_e^{\text{AC}}$	input (α_e^{AC} only)	hinted (S1)	hinted
Module-to-inverter assignment $s_{m,m'}$	decided	hinted (as $s_{m,k}$)	hinted
PV strings $s_{m,k}, f_{e,k}, L_{m,k}$	—	hinted	hinted
DC homeruns, and trenches $h_{m,k}, z_{e,k}^{\pm}, \phi_e, \tau_e^{\text{DC}}$	—	decided	hinted

Table 6.1: Variable handover across stages. “(S1)” and “(S2)” annotate the source stage where it is not the immediately preceding one.

6.6 Evaluation Against the Human Annotation

The freezing study of Section 5.3 ended with a verdict: past a few hundred modules the monolithic model returns no layout at all unless the PV-strings, the inverter placement, and the string-to-inverter assignment are fixed to hand-annotated values. The four-stage decomposition of this chapter constructs exactly those decisions automatically. The evaluation asks how good that construction is: given the same plant, does the automatic layout cost less—and lay less cable—than the design an engineer drew by hand? The question is Chapter 1’s, made concrete: the annotation follows the sequential place–string–route workflow that Section 1.1 identified as the industry default, and the staged decomposition is the integrated alternative this work set out to build.

The comparison starts with the objective cost. The baseline prices the complete human annotation: the engineer’s decisions are pinned into the Chapter 4 model, and the solver completes only what the annotation leaves open—the DC-homerun and AC-feeder routes. The margin between human annotated

and computed layout, however, carries no currency: the coefficients of Section 5.1 are not market prices but a deliberate ordering of magnitudes, so a cost difference identifies the cheaper design under that ordering and nothing more. The evaluation therefore reports the objective totals first and then decomposes every difference into the physical quantities behind it—metres of cable and trench, and inverter counts—which do not depend on the coefficients.

6.6.1 Protocol

The instances are the nine real-world solar plants of Table 5.5, labelled P1–P9 by module count. The baseline freeze extends the deepest configuration of the freezing study (F_3 in Table 5.6): the PV-strings with their terminals and the inverter placement are frozen to the engineer’s values as in F_3 , and in addition the string-to-inverter assignment is clamped and the DC trench layout is pinned to the annotated trenches. The Chapter 4 solver then completes the only decisions the annotation leaves open—the DC-homerun and AC-feeder routes—within a 1800s budget. This residual solve reaches optimality on P1–P4 and ends with a gap below 0.17% on the other five plants, so the baseline routes are, up to that margin, the shortest the annotation admits.

The staged decomposition Chapter 6 receives no annotation and decides every variable itself, inside a 7200s budget split across its four stages, the same on every instance; the per-stage slices, model sizes, and terminal gaps are tabulated in Appendix F.3. Both designs route on the same DC and AC routing graphs. Every configuration ran once, under solver seed 42.

Stage 1 prices cabling as a single charge on the aggregated arc flow, Stage 2 prices only PV-string cable and trench, and only Stages 3 and 4 score the complete Chapter 4 objective (the per-stage objectives and their coefficients are spelled out in Appendix F.3).

6.6.2 Total Cost

The staged decomposition Chapter 6 produces a complete layout on all nine plants, where the unfrozen monolithic model yielded an incumbent on one (Section 5.3). Every one of those layouts is cheaper than the human annotated layout. Figure 6.2 shows the totals; the term-by-term breakdown behind them is tabulated in Table F.1. The staged decomposition undercuts the baseline on all nine instances, by 0.3% (P2) to 5.2% (P7 and P8). Where each margin comes from—shorter routes or less inverters—the totals do not say; Section 6.6.3 decomposes them into the metres and counts behind the figure.

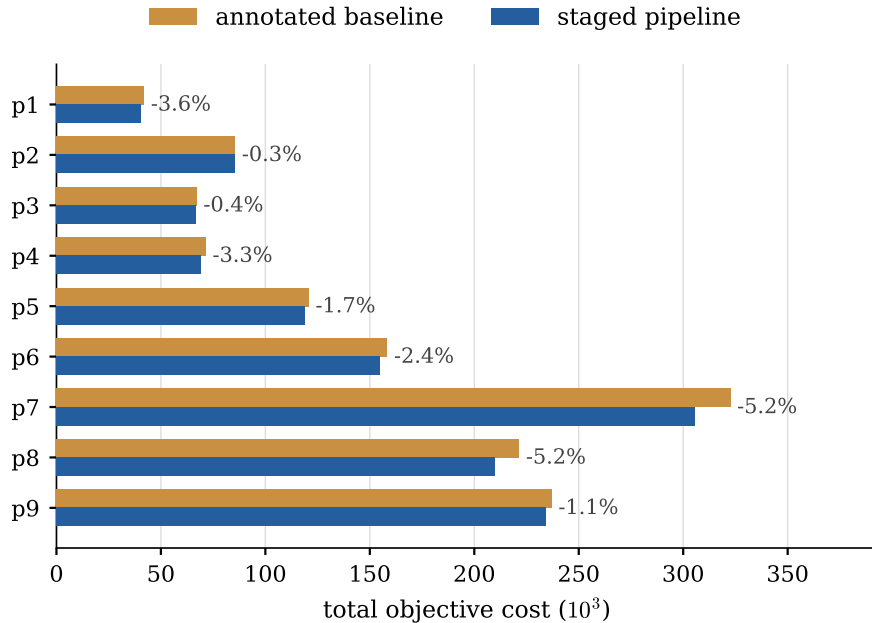


Figure 6.2: Total objective cost of both designs on each instance; each label gives the staged total relative to the baseline. The staged design is cheaper on all nine plants. The term-by-term breakdown of each difference is tabulated in Table F.1.

6.6.3 Cable Lengths and Inverter Counts

Figure 6.3 reports the PV-string, DC-homerun, and AC-feeder cable lengths, the DC-trench length, and the inverter count on every instance; Appendix F.2 tabulates the exact figures.

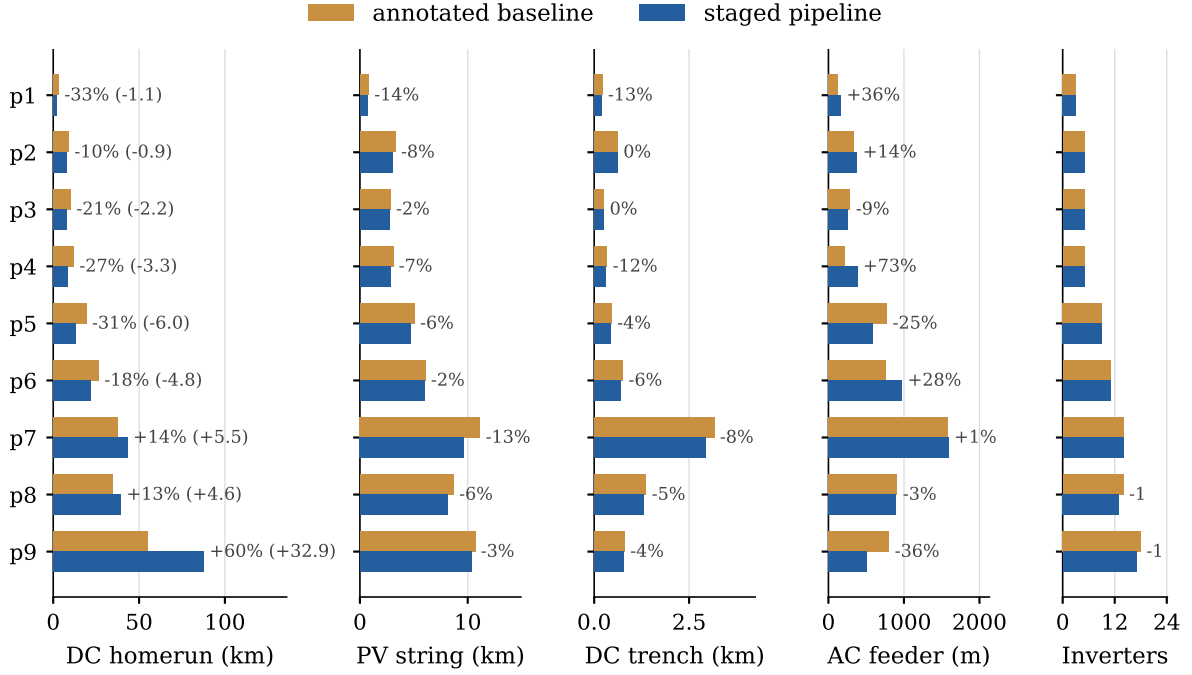


Figure 6.3: Annotated baseline against staged decomposition on each instance: the PV-string, DC-homerun, and AC-feeder cable lengths, the DC-trench length, and the inverter count. Each label gives the staged value relative to the baseline; the length labels add the absolute difference in kilometres or metres, the inverter labels the count difference. The staged decomposition never lays more PV-string cable or trench.

Read across the nine instances, Figure 6.3 separates the wins into two regimes. On P1–P6—the *routing regime*—the two designs buy the same inverter type, with string and inverter counts matching on every plant, so the staged decomposition’s entire margin is routing quality: less homerun and PV-string cable on each instance, and never more trench. On P7–P9—the *trade regime*—the relation inverts: the staged homeruns are *longer*, and the added metres are the price of a redesign, buying a sparser trench network on P7 and one inverter fewer on P8 and P9. The paragraphs below unpack the two regimes in turn; Table F.1 carries the term-by-term cost arithmetic behind each.

The staged decomposition out-routes the human annotation layout on six of nine plants. On P1–P6 the staged layout lays 10–34% less homerun cable—18.3 km saved over the six plants—with the same string and inverter counts and never more trench: the saving is pure routing quality. Since the baseline routes are already the shortest the annotation admits (Section 6.6.1), the saved kilometres do not come from better route-finding but from better stringing: the string composition, terminal placement, and string-to-inverter assignment that the annotation fixes up front, the staged decomposition optimises jointly with the trench layout. The PV-string cable shows the same signature—2–14% shorter on every instance.

AC feeders grow only where they buy homerun savings. Summed over the nine plants, both designs lay about 5.8 km of feeder cable, within one percent of each other, but the per-instance differences have a direction. On P1, P4, and P6 the staged feeders are 28–73% longer, and Table F.2 shows what the extra metres buy: on the same plants the homeruns shorten by kilometres—on P4, 162 m of added feeder against 3.3 km of saved homerun. The trade is inverter placement (Appendix A): an inverter pulled toward its strings shortens their homeruns and lengthens its feeder, and with an AC metre priced at ten homerun metres ($c_e^{\text{ac}} = 1$, $c_e^{\text{homerun}} = 0.1$) the exchange pays whenever one added feeder metre saves more than ten homerun metres—here it saves about twenty.

The three largest plants trade homerun cable for inverters and trenches. On P7–P9 the staged homeruns are *longer*, by 14%, 13%, and 60%—the visible half of a trade: under the cost ordering of Section 5.1, homerun cable is the cheapest quantity to stretch whenever stretching it removes an inverter or a trench. P7 concentrates its circuits onto a sparser trench backbone; P8 and P9 each remove one inverter and redistribute its strings over the remaining ones. In all three cases the saving repays the added cable.

The layouts look alike; the metres differ. Figure 6.4 sets the two P5 designs side by side: the same module field, 90 strings, 9 inverters, the same central trench corridor—at plant scale they are hard to tell apart. Yet 6.0 km of homerun cable separates them, hidden in terminal orientations and string-to-inverter pairings the eye does not resolve. Layout quality at this scale is invisible in a drawing; only the length accounting reveals it. Appendix F.4 shows all nine pairs.

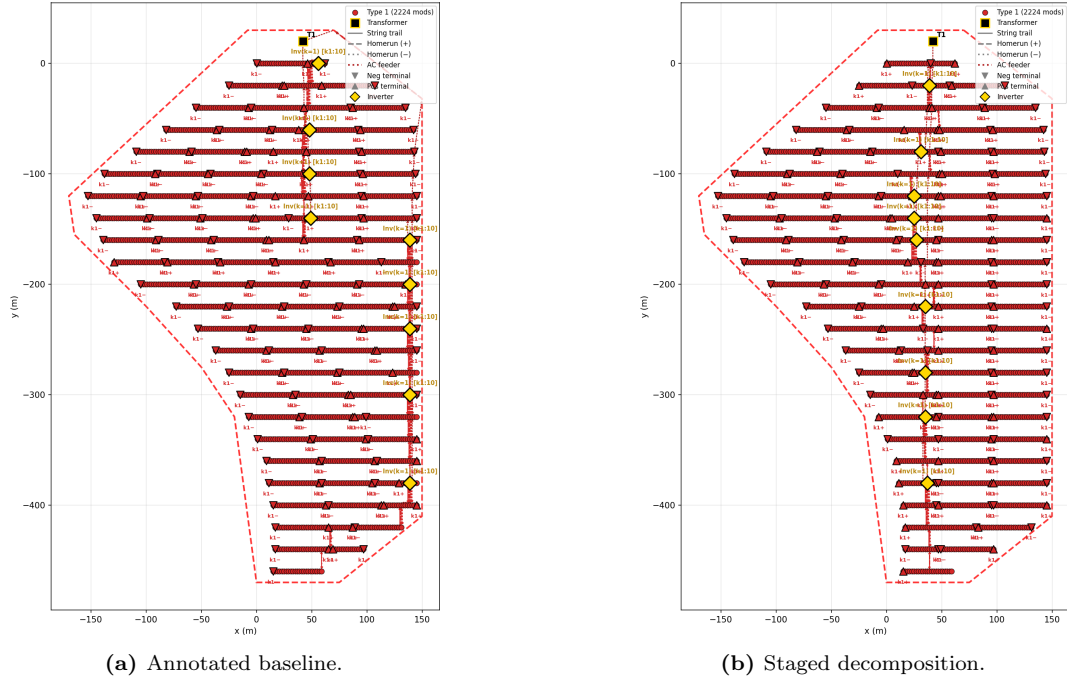


Figure 6.4: The P5 layout (2224 modules) under both designs: 90 strings, 9 inverters, and one transformer in each. The designs are visually interchangeable, yet kilometres of cable separate them (Figure 6.3): the difference sits in terminal and assignment choices the drawing does not resolve.

What the evaluation establishes. On nine distributed-inverter sites, the staged decomposition replaces the annotation: it produces a layout on every plant, each cheaper under the model’s objective. On the six smaller plants it delivers the engineer’s design with 10–34% less DC-homerun cable, part of it bought with longer AC feeders; on the three largest it restructures the design, trading cheap cable for inverters and trenches—an exchange the sequential workflow cannot make. The evidence has limits: one reference design per plant, a single seeded run, a cost ordering rather than market prices.

Chapter 7

Conclusion

Chapter 1 asked whether the six design decisions — string partition; inverter type, location, and assignment; DC-homerun and AC-feeder routing — can be made jointly, by one optimisation model, on distributed-inverter plants laid out on irregular real-world sites. That question has an affirmative answer, carried not by the monolithic programs of Chapters 3 and 4 but by the four-stage decomposition of Chapter 6. Section 7.1 develops that answer, and Section 7.2 states the boundaries within which it holds.

7.1 Summary of Findings

This work asked whether the six core decisions of a distributed-inverter plant design—string partition; inverter type, location, and assignment; DC-homerun and AC-feeder routing—can be made jointly, by one optimisation model, on the irregular small-to-medium sites of new European deployment (Chapter 1). No prior model was both fully scoped over the six decisions and solved as a single program: integrated optimisation had been demonstrated only for central-inverter plants on rectangular districts [6], and the one full-scope predecessor reaches its design across three sequential stages [11]. The answer is affirmative, subject to one methodological qualification: at the scale of real-world solar plants the monolithic programs are computationally out of reach—of the nine benchmark plants, the stronger of the two yields a layout on one within an hour, and that layout still carries a 10.5% gap—and the joint design becomes tractable only through a decomposition that fixes the trench topology before the PV-strings. Three artefacts substantiate this conclusion: a pair of monolithic MILP formulations, a benchmark of nine annotated real-world solar plants, and a four-stage decomposition evaluated against that benchmark.

Two monolithic MILP formulations give the question of Chapter 2 a solvable form. The string-indexed model of Chapter 3 carries one set of variables per PV-string and reconstructs every DC circuit exactly. The inverter-type-indexed model of Chapter 4 aggregates strings by inverter type, so its size depends on the inverter types rather than on the module count; the aggregation gives up exact circuit reconstruction, and the DC circuit-length limit then holds only per inverter on average.

Only the inverter-type-indexed model scales. Under the single-type catalogues, at $N_A = 32$ arrays the string-indexed model is 12 times the type-indexed model by constraint count and more than 300 times by nonzeros; at $N_A = 64$ it exhausts the time budget or the 32 GB of the test machine on every run. Its one structural advantage, an LP relaxation tighter by 0.45–1.20%, is out of reach: the relaxation alone overruns the time budget from $N_A = 32$ onward. The inverter-type-indexed model holds its gap below 0.04% through $N_A = 128$ under a single-type, single-transformer catalogue, and the scaling study of Chapter 5 shows that catalogue richness, not plant size, is the harder axis: under a three-type catalogue the same model degrades from gaps of 6–11% at 8–32 arrays to no incumbent at $N_A = 128$.

The empirical claims are grounded in nine real-world solar plants of 313 to 4889 modules, each transcribed from an engineering drawing and annotated by hand with an engineer-style reference design; published with the code, they are a benchmark against which future layout algorithms can be measured. On them the failure mode of the monolithic approach is feasibility, not optimality: within the hour budget the unfrozen inverter-type-indexed model returns no incumbent on eight of the nine plants, and on the ninth—the smallest—its incumbent still carries a 10.5% gap at timeout. The freezing study of Section 5.3 located the difficulty. Fixing the inverter placement alone recovers only three plants, two of them with gaps above 20%; fixing the PV-strings restores feasibility on the six smaller plants; fixing strings and placement together yields incumbents on all nine, with gaps below 0.7% everywhere except P7 and a first incumbent within 13 seconds on eight of the nine. String formation dominates the difficulty, placement

matters through its coupling with the strings, and the residual assignment and routing problem is easy.

The four-stage solver of Chapter 6 is that diagnosis made algorithmic: it fixes the topology—inverter placements, DC trenches, transformer assignments, and AC feeders—on an aggregated single-commodity flow, resolves module-to-inverter paths as a multi-commodity flow, re-solves the full inverter-type-indexed model under the frozen topology, and releases the freeze for a final polish. Receiving no annotation, it produces a complete layout on all nine plants—where the monolithic model produced one—and every layout prices 0.3–5.2% below the near-optimally routed annotation under the model’s own objective.

The margin has two regimes. On P1–P6 the staged decomposition delivers the human annotated design’s exact inverter and string counts and wins on routing alone: 10–34% less DC-homerun cable, an aggregate saving of 18.3 km; PV-string cable 2–14% shorter on every instance; a trench network nowhere longer; and AC feeders that lengthen on four of the six plants to buy larger homerun savings, yet summed over all nine plants end about one percent shorter than the annotation’s. On P7–P9 the staged decomposition revises the design instead: it accepts longer homerun cable to eliminate seven trenches on P7 and one inverter on each of P8 and P9. The decomposition succeeds by keeping these two decisions—how many inverters to place and how to form the strings—inside the optimisation, priced jointly with the cable each choice implies rather than fixed up front, and the cable it saves against a near-optimally routed human annotated design is the measured value of that freedom.

7.2 Limitations

The findings hold within five boundaries, set by the modelling choices of Chapter 2 and the experimental design of Chapters 5 and 6.

The cost coefficients fix an ordering, not prices. The coefficients of Section 5.1 fix which quantities are expensive relative to which—trenching above cabling (Appendix A), inverters above both (Table 5.1)—not what any of them costs in currency. The cost comparison of Section 6.6.2 therefore certifies the staged layouts cheaper only under that assumed ordering, and every further comparison is stated in metres and unit counts.

The staged decomposition carries no optimality certificate. The staged solver is a heuristic. Stages 1 and 2 optimise surrogate objectives, so their terminal gaps—13–78% for Stage 1, up to 94% for Stage 2 and unreported on two plants—bound nothing about the full model, and Stage 4, the only stage that sees the complete model, ends its budget at gaps of 12–57% while improving the Stage 3 incumbent only marginally (Appendix F.3). The layouts are therefore certified only against the human annotation, never against the optimum; how much better a layout the full model admits on these plants is unknown.

The transcribed site geometry approximates the real-world solar plant. Each instance is transcribed by hand from an engineering drawing into the site model of Section 2.3: the arrays, their modules, and their relative arrangement follow the drawing, but the distances between them are placed by eye, not measured.

The electrical engineering is carried only as lengths and counts. The model enforces the plant’s electrical constraints in the abstracted form of Chapter 2: string sizes and inverter loads as counts, and voltage drop as the single cap $L_{DC}^{\max}$ on each DC-circuit length (Section 2.3). Cable cross-sections, actual voltage and current levels, resistive losses, and protection equipment sit outside the model, so a layout optimal in metres and counts still requires electrical validation before it is buildable.

The evidence base is nine plants and one run each. Every empirical claim about the staged decomposition rests on the nine benchmark instances, each solved once, under a single solver seed and a fixed 7 200 s budget (Section 6.6.1). How the margins vary across seeds, across time budgets, or on plants outside the 1–10 MW range is not measured.

7.3 Statement on the Use of AI Tools

Large language models were used during the preparation of this thesis to polish prose and fix grammar in the writing, and as a coding assistant during the implementation of the solvers, and experiment scripts. All such output was reviewed and validated by the author before use. The ideas, mathematical formulations, and experimental designs presented in this thesis are the author's own.

Bibliography

- [1] International Renewable Energy Agency. Renewable capacity statistics 2026. Technical report, IRENA, Abu Dhabi, 2026.
- [2] International Energy Agency. Renewables 2024: Analysis and forecast to 2030. Technical report, IEA, Paris, 2024.
- [3] International Renewable Energy Agency. Renewable power generation costs in 2024. Technical report, IRENA, Abu Dhabi, 2025.
- [4] David Feldman, Vignesh Ramasamy, Michael Woodhouse, Jarett Zuboy, Jal Desai, and Andy Walker. U.S. solar photovoltaic system and energy storage cost benchmarks, with minimum sustainable price analysis: Q1 2024. Technical Report NREL/TP-7A40-92536, National Renewable Energy Laboratory, Golden, CO, 2024.
- [5] Tekai Eddine Khalil Zidane, Ali Saleh Aziz, Younes Zahraoui, Hossam Kotb, Kareem M. AboRas, Kitmo, and Yosef Berhan Jember. Grid-connected solar pv power plants optimization: A review. *IEEE Access*, 11:79588–79608, 2023.
- [6] Chong Wang, Qinghua Wu, Kai Pan, and Zuo-Jun Max Shen. Layout optimization for a large-scale grid-connected solar power plant. *INFORMS Journal on Computing*, 37(6):1560–1586, 2024.
- [7] SolarPower Europe. EU solar market outlook 2025–2030. Technical report, SolarPower Europe, Brussels, 2025.
- [8] Sascha Gritzbach, Dominik Stampa, and Matthias Wolf. Solar farm cable layout optimization as a graph problem. *Energy Informatics*, 5(1):25, 2022.
- [9] Thomas Bläsius, Max Göttlicher, Sascha Gritzbach, and Wendy Yi. The constrained layer tree problem and applications to solar farm cabling. In *2025 Proceedings of the Symposium on Algorithm Engineering and Experiments (ALENEX). Hrsg.: R. Chowdhury, J. Berry, K. Hanauer, B. Ren*, volume 2025-January of *Proceedings*, page 147 – 159. Society for Industrial and Applied Mathematics Publications (SIAM), 2025.
- [10] Lara Löhken and Michael Stiglmayr. A multi-objective perspective on the cable-trench problem. *Journal of Combinatorial Optimization*, 49(4):55, 2025.
- [11] Juan Carlos Cortez, Jéssica A. A. Silva, Rody A. Gallegos H., Marina Lavorato, and Marcos J. Rider. Three-stage optimization-based framework for grid-connected sizing of a pv power plant system. *IEEE Journal of Photovoltaics*, 15(6):896–907, 2025.
- [12] Tamer T. N. Khatib and Hiba Assi. A novel method for optimizing grid-connected photovoltaic power plant layouts considering solar inverter location and power cables paths. *Renewable Energies*, June 2025.
- [13] Meng Zhang, Jixuan Wei, Rong Tang, Qin Hu, Yang Wang, Li Chang, Xingcheng Gan, and Ji Pei. Enhanced swarm-intelligence optimization of inverter placement for cable cost minimization in standardized photovoltaic power units. *Energies*, 18(19), 2025.
- [14] Yuanqing Yao, Yibo Wang, Hongjie Jia, and Yunfei Mu. A novel differential evolution method with a hierarchical decoder for the photovoltaic layout optimization problem. *International Journal of Electrical Power & Energy Systems*, 164:110397, 2025.

- [15] Bezalel Gavish and Stephen C Graves. The travelling salesman problem and related problems. 1978.
- [16] Bezalel Gavish. Formulations and algorithms for the capacitated minimal directed tree problem. *Journal of the ACM (JACM)*, 30(1):118–132, 1983.

Appendix A

Choice of cost coefficients

This appendix details the rationale behind the default cost coefficients of Table 5.1.

Three relations among the cabling and trenching coefficients motivate the choice. First, $c_e^{\text{pv}} > c_e^{\text{home}}$: the PV-string cable is the dominant intra-string cost, so the solver is encouraged to form strings that follow the within-array geometry tightly before reaching for DC-homerun cable to close the circuit back to the inverter. Second, c_e^{trench} is the largest of the cable- and trenching-related coefficients: a trench, once dug, accommodates up to $N_{\text{DC}}^{\text{trench}}$ cables, so making it expensive consolidates routing into a small number of shared cross-array corridors rather than dispersing each cable into its own trench. Third, c_e^{home} and c_e^{ac} are kept comparable enough to expose the trade-off that inverter placement induces between them. Moving an inverter toward the transformer shortens its AC feeders but lengthens its DC homeruns; moving it toward its assigned strings does the reverse. Without comparable coefficients the solver would degenerate into one of two extremes—inverters clustered next to the transformer to minimise AC routing, or buried among their strings to minimise DC routing—rather than finding a placement that serves both.

The overlap penalties c^{ov} and c_{ω}^{tov} are calibrated against the AC trench cost c_e^{ac} so that each translates into an equivalent length of AC trench. With $c_e^{\text{ac}} = 1$ per metre, a penalty of 200 equals 200 metres of trench: the solver accepts an overlap precisely when avoiding it would require a detour of more than 200 metres. Short detours are therefore preferred, while an overlap that geometry makes unavoidable is tolerated rather than punished prohibitively.

Appendix B

Synthetic instance geometry

This appendix gives the full configuration of the synthetic instance generator of Section 5.2.1 (Table B.1) and the rationale behind its capacity bounds.

Constant	Value	Meaning
Modules per array	24	one PV string per array
Module spacing	2.0 m	module pitch along x
Array spacing	10.0 m	row pitch along y
Transformer offset	15.0 m	vertical offset of transformers below row 0
Boundary padding	5.0 m	axis-aligned rectangle around the module-transformer extent
$L_{\text{radius}}^{\text{DC}} = L_{\text{radius}}^{\text{AC}}$	20.0 m	DC and AC search radius
$L_{\text{DC}}^{\text{max}}$	2000.0 m	upper bound on the total DC-circuit length
$N_{\text{DC}}^{\text{trench}}$	32	maximum DC cables sharing a single cross-array trench
$N_{\text{AC}}^{\text{trench}}$	4	maximum AC-feeder cables sharing a single AC trench

Table B.1: Configuration constants of the synthetic instance generator. All sizes share the same per-array geometry and the same capacity bounds; increasing N_A simply stacks more rows.

The DC circuit-length bound $L_{\text{DC}}^{\text{max}} = 2000$ m is a modeling safeguard, sized generously from the synthetic geometry rather than from a real-world ceiling. A single DC circuit consists of one PV-string—roughly 24 modules, or about 46 m at the 2 m module pitch—closed by the two DC-homeruns that return its terminals to the inverter. A budget of 2000 m leaves room for both homeruns to reach across even the largest instance, together with the detours that shared DC-trenches impose. The layouts the solver selects use far shorter circuits, so the bound shapes none of the solutions in this study; it serves only to exclude degenerate, arbitrarily long circuits. The two trench-capacity values, $N_{\text{DC}}^{\text{trench}} = 32$ and $N_{\text{AC}}^{\text{trench}} = 4$, are fixed configuration choices of the generator, held constant across all instance sizes.

Appendix C

Scaling study detail

This appendix details the model-size summary of Section 5.2.3 and lists the per-cell outcomes behind Section 5.2.5.

C.1 Constraint growth by family

Every constraint family is generated over an index set—strings, inverter types, routing arcs, or pairs thereof—so its count simply follows the growth of that set. Table C.1 makes this concrete at $N_A = 32$. In the string-indexed formulation the four dominant DC-side families—PV-string formation, topological routing, string-to-inverter assignment, and the DC homeruns—couple a per-string variable to every routing arc; the number of strings and the number of arcs each grow linearly with N_A , so these families grow quadratically and contribute 92% of all constraints. The inverter-type-indexed formulation indexes the same families by inverter type instead of by string, so each grows linearly, scaled by the size of the type set $\mathcal{K}$ (the Scenario B columns). The AC-side families are shared by both formulations and grow linearly throughout.

Constraint family	String-indexed (Ch3)				Inverter-type-indexed (Ch4)			
	Constraints	A	B	C	Constraints	A	B	C
<i>DC domain (on the DC graph)</i>								
PV-string formation	C3-C3.4	98 368	98 368	98 368	C3-C4.5	3 073	7 683	3 073
Topological routing	C4.1-C4.6	193 056	193 056	193 056	C4.1-C4.4	6 032	18 096	6 032
String-to-inverter	C5-C6.5	50 816	50 816	50 816	C5-C6	2 305	5 377	2 305
DC homeruns	C7.1-C7.3	119 296	119 296	119 296	C7.1, C7.2, C8.1	4 496	7 568	4 496
DC circuit length	C7.4	32	32	32	C8.2-C8.4	4 496	4 496	4 496
DC trenching	C13.1-C13.3	2 232	2 232	2 232	C13.1-C13.3	2 232	2 232	2 232
DC-domain total		463 800	463 800	463 800		22 634	45 452	22 634
<i>AC domain (on the AC graph)</i>								
Inverter-to-transformer	C8.1-C11	26 913	26 913	27 682	C9.1-C11.2	4 610	7 684	5 379
AC feeders	C12.1-C12.5	9 707	9 707	10 502	C12.1-C12.5	9 707	9 707	10 500
AC/DC overlaps	C14-C15	1 404	1 404	1 404	C14-C15	1 404	1 404	1 404
AC-domain total		38 024	38 024	39 588		15 721	18 795	17 283
<i>Placement (geometry-bound)</i>								
Inverter placement	C1-C2.3	773	777	773	C1-C2.3	773	777	773

Table C.1: Constraint families grouped by domain, at $N_A = 32$ for the three scenarios (A: single transformer and inverter type; B: three inverter types; C: two transformers). The *Constraints* columns give the constraint-tag ranges that compose each family—the constraint sections of Chapters 3 and 4—and the *A*, *B*, *C* columns the number of constraints it contributes under each scenario. The string-indexed counts are scenario-invariant in the DC domain and rise under Scenario C in the AC domain; the inverter-type-indexed counts inflate under the three inverter types of Scenario B. The DC-domain families are built on the DC graph and the AC-domain families on the AC graph; the per-domain totals are the aggregates plotted against the edge counts in Figure C.1. Inverter placement is geometry-bound and not edge-matched.

Figure C.1 plots the same counts against the size of the routing graphs. Because the generator stacks identical rows, every dimension of the instance graph—DC edges, AC edges, and the conflict sets—grows linearly with the plant, so the figure isolates the formulation’s contribution: the inverter-type-indexed count stays at a near-constant number of constraints per edge, whereas the string-indexed count rises super-linearly in the edge count.

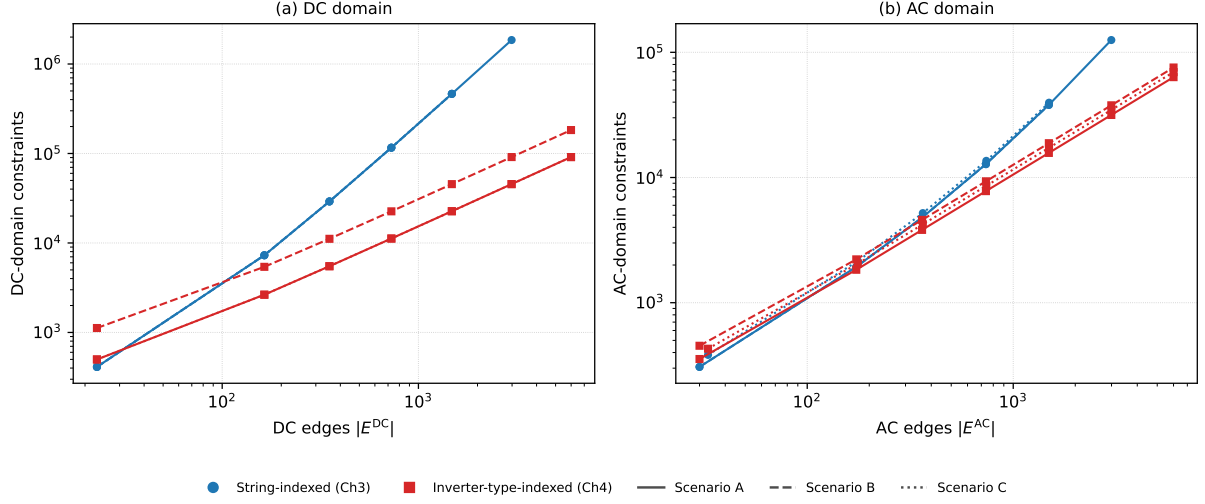


Figure C.1: Constraint count against graph size for the three scenarios (log-log), domain-matched: the DC-domain families against the DC edge count $|E^{DC}|$ (left) and the AC-domain families against the AC edge count $|E^{AC}|$ (right). Formulation is shown by colour and marker, scenario by line style. The string-indexed counts grow super-linearly in the edge count and are scenario-invariant; the inverter-type-indexed counts grow linearly, with the three-type catalogue of Scenario B sitting highest.

C.2 Per-cell results

Table C.2 reports every cell of the scaling study: status, solve time, the bounds at termination, the optimality gap, and the model size, aggregated over the three solver seeds.

Table C.2: Per-cell detail for the scaling study, aggregated over the n seeds. Solve time t (Gurobi Runtime) and MIP gap are reported as mean $\pm$ standard deviation; the lower bound (LB, the dual bound at termination) and upper bound (UB, the best incumbent) are medians over seeds; variables, constraints, and constraint-matrix nonzeros are single values, as the model build is deterministic. A dash marks a quantity with no value (e.g. no incumbent leaves no UB or gap). OOM marks a seed on which the solver exhausted the machine’s memory; the string-indexed formulation was not attempted at $N_A = 128$.

Solver	Scn	N_A	n	Status	t (s)	LB	UB	Gap (%)	Vars	Constrs	Nonzeros
Ch3	A	1	3	opt	0.02 ± 0.00	10 065	10 066	0.000 ± 0.000	528	748	3 653
Ch3	A	4	3	opt	13.53 ± 8.29	40 243	40 247	0.009 ± 0.000	8 310	9 321	128 300
Ch3	A	8	3	opt(2), TL(1)	586.43 ± 531.47	80 527	80 535	0.873 ± 1.494	31 946	34 093	878 176
Ch3	A	16	3	TL	1200.13 ± 0.02	161 125	371 448	55.984 ± 1.230	124 626	129 333	6 452 168
Ch3	A	32	3	TL	1801.21 ± 1.77	320 120	—	—	491 618	502 597	49 379 992
Ch3	A	64	3	TL(2), OOM(1)	2710.91 ± 2.10	640 241	—	—	1 952 130	1 980 261	386 218 040
Ch3	B	1	3	opt	0.02 ± 0.00	10 065	10 066	0.000 ± 0.000	578	752	4 139
Ch3	B	4	3	TL	1200.05 ± 0.02	16 247	16 729	3.182 ± 1.051	8 504	9 325	131 378
Ch3	B	8	3	TL	1200.10 ± 0.02	30 473	63 940	51.605 ± 23.261	32 332	34 097	887 398
Ch3	B	16	3	TL	1200.11 ± 0.01	60 841	358 131	82.905 ± 0.336	125 396	129 337	6 482 894
Ch3	B	32	3	TL	1800.85 ± 1.18	121 826	—	—	493 156	502 601	49 490 590
Ch3	B	64	3	OOM	—	—	—	—	—	—	—
Ch3	C	1	3	inf	0.01 ± 0.00	—	—	—	657	824	4 206
Ch3	C	4	3	opt	32.03 ± 2.77	40 269	40 272	0.009 ± 0.000	8 869	9 541	130 787
Ch3	C	8	3	TL(2), opt(1)	1157.74 ± 73.42	80 498	80 633	3.841 ± 6.499	33 073	34 505	883 215
Ch3	C	16	3	TL	1200.13 ± 0.02	161 108	374 095	56.834 ± 0.230	126 889	130 129	6 462 311
Ch3	C	32	3	TL	1800.41 ± 0.42	322 845	—	—	496 153	504 161	49 400 343
Ch3	C	64	3	OOM	—	—	—	—	—	—	—
Ch4	A	1	3	opt	0.02 ± 0.00	10 065	10 066	0.008 ± 0.000	715	888	2 804
Ch4	A	4	3	opt	3.47 ± 2.12	40 243	40 247	0.009 ± 0.001	4 246	4 581	16 037
Ch4	A	8	3	opt	368.82 ± 107.17	80 527	80 535	0.010 ± 0.000	8 970	9 517	33 809
Ch4	A	16	3	TL	1200.04 ± 0.02	161 201	161 230	0.018 ± 0.001	18 418	19 389	69 353
Ch4	A	32	3	TL	1800.12 ± 0.04	323 025	323 099	0.023 ± 0.000	37 314	39 133	140 441
Ch4	A	64	3	TL	2700.07 ± 0.01	648 578	648 761	0.028 ± 0.000	75 106	78 621	282 617
Ch4	A	128	3	TL	3600.23 ± 0.08	1 307 331	1 307 762	0.032 ± 0.001	150 690	157 597	566 969
Ch4	B	1	3	opt	0.05 ± 0.00	10 065	10 066	0.008 ± 0.001	1 421	1 620	6 168
Ch4	B	4	3	opt	281.09 ± 36.99	16 728	16 729	0.008 ± 0.004	8 216	7 749	33 405
Ch4	B	8	3	TL	1200.05 ± 0.02	31 550	33 482	6.175 ± 0.749	17 292	15 933	69 785
Ch4	B	16	3	TL	1200.10 ± 0.02	60 995	66 994	8.928 ± 0.083	35 444	32 301	142 545
Ch4	B	32	3	TL	1800.12 ± 0.05	121 927	136 734	11.192 ± 2.086	71 748	65 037	288 065
Ch4	B	64	3	TL	2700.20 ± 0.03	243 643	781 718	68.577 ± 4.183	144 356	130 509	579 105
Ch4	B	128	3	TL	3600.05 ± 0.01	485 580	—	—	289 572	261 453	1 161 185
Ch4	C	1	3	inf	0.00 ± 0.00	—	—	—	844	962	3 349
Ch4	C	4	3	opt	203.32 ± 63.40	40 269	40 272	0.009 ± 0.000	4 805	4 799	18 516
Ch4	C	8	3	TL(2), opt(1)	1139.65 ± 104.59	80 524	80 535	0.014 ± 0.004	10 097	9 927	38 840
Ch4	C	16	3	TL	1200.08 ± 0.04	161 183	161 230	0.029 ± 0.001	20 681	20 183	79 488
Ch4	C	32	3	TL	1800.07 ± 0.01	322 999	323 099	0.032 ± 0.002	41 849	40 695	160 784
Ch4	C	64	3	TL	2700.11 ± 0.06	648 453	649 439	0.185 ± 0.138	84 185	81 719	323 376
Ch4	C	128	3	TL	3600.87 ± 0.94	1 306 706	1 320 233	1.785 ± 2.042	168 857	163 767	648 560

Appendix D

Freezing study detail

This appendix lists the model dimensions of the nine real instances and the per-cell solver outcomes behind Section 5.3.3.

D.1 Model size on the real instances

Table D.1 reports the dimensions of the inverter-type-indexed model on each instance. A freeze does not change these dimensions: frozen variables are fixed through their bounds and removed by presolve, so all four configurations of an instance start from the same build.

Table D.1: Model size of the inverter-type-indexed formulation on the nine real instances. Freezing fixes variables through their bounds, so all four configurations of an instance share the model dimensions listed here; presolve then removes the fixed part.

Inst.	Variables	Binaries	Constraints	Nonzeros
P1	13 190	3 218	14 505	47 796
P2	102 063	22 837	93 299	379 500
P3	68 575	16 704	70 787	248 145
P4	67 546	16 519	70 348	244 662
P5	119 197	29 121	122 385	432 429
P6	153 031	37 957	155 175	556 051
P7	374 674	82 051	322 987	1 442 532
P8	361 291	75 149	312 062	1 388 956
P9	486 098	101 052	423 795	1 872 145

D.2 Per-cell solver outcomes

Table D.2 lists every (instance, configuration) cell. Two protocol details complement Section 5.3.2. First, the F_0 baseline runs under solver seed 42 and the freeze runs under the solver default. Second, all four configurations drop the spanning-tree cap that would limit the trenched cross-array edges to a tree: the baseline and the freezes solve the same model up to the frozen variables, so the freeze outcomes are not influenced by the cap.

Table D.2: Per-cell detail for the freezing study. Solve time t is Gurobi **Runtime**; LB is the dual bound and UB the best incumbent at termination; the root-LP gap is the relative gap between the final incumbent and the root relaxation; t_{inc} is the time to the first incumbent. A dash marks a quantity with no value: a solve without an incumbent leaves no UB, gap, root gap, or t_{inc} . Status *opt* is an optimality certificate at the 10^{-4} gap target; *TL* is the 3600 s time limit.

Inst.	Config	Status	t (s)	LB	UB	Gap (%)	Root gap (%)	Nodes	t_{inc} (s)
P1	F_0	TL	3600.1	36 225	40 482	10.5	25.59	27 699	20.9
	F_1	TL	3600.0	39 046	40 466	3.5	21.72	137 401	11.1
	F_2	TL	3600.0	41 918	41 945	0.07	0.48	386 144	0.5
	F_3	opt	0.6	41 996	42 000	0.01	0.12	1	0.2
P2	F_0	TL	3600.3	52 194	—	—	—	1	—
	F_1	TL	3600.1	60 715	—	—	—	1 302	—
	F_2	TL	3600.3	82 580	85 491	3.4	4.36	6 805	66.8
	F_3	opt	2720.9	85 544	85 551	0.01	2.29	42 688	4.4
P3	F_0	TL	3600.1	52 806	—	—	—	3 049	—
	F_1	TL	3600.2	57 811	86 163	32.9	35.79	6 818	3144.0
	F_2	TL	3602.5	65 772	67 032	1.9	2.51	7 143	17.1
	F_3	opt	368.0	67 145	67 147	0.00	1.09	50 198	2.4
P4	F_0	TL	3600.2	54 126	—	—	—	1 305	—
	F_1	TL	3600.1	58 425	75 496	22.6	27.33	7 244	2745.0
	F_2	TL	3600.1	69 739	71 445	2.4	2.67	7 133	38.1
	F_3	opt	278.0	71 499	71 506	0.01	1.51	35 039	2.4
P5	F_0	TL	3600.2	94 951	—	—	—	1 315	—
	F_1	TL	3600.2	103 392	—	—	—	6 437	—
	F_2	TL	3607.1	118 688	120 596	1.6	2.07	6 659	80.4
	F_3	TL	3600.1	120 552	120 796	0.20	0.72	520 339	6.6
P6	F_0	TL	3619.2	116 517	—	—	—	1	—
	F_1	TL	3600.2	128 247	—	—	—	1 093	—
	F_2	TL	3601.0	154 550	157 252	1.7	4.30	7 546	765.9
	F_3	TL	3600.5	156 962	157 381	0.27	0.80	95 454	11.9
P7	F_0	TL	3600.1	130 847	—	—	—	1	—
	F_1	TL	3600.1	172 455	—	—	—	19	—
	F_2	TL	3600.1	300 766	—	—	—	32	—
	F_3	TL	3600.2	306 823	314 645	2.5	3.06	7 182	12.2
P8	F_0	TL	3600.1	130 632	—	—	—	1	—
	F_1	TL	3600.1	162 710	—	—	—	4	—
	F_2	TL	3600.1	215 212	—	—	—	20	—
	F_3	TL	3600.1	220 990	221 511	0.24	1.49	28 053	6.8
P9	F_0	TL	3600.1	170 867	—	—	—	1	—
	F_1	TL	3600.1	202 222	—	—	—	1	—
	F_2	TL	3600.1	229 864	—	—	—	1	—
	F_3	TL	3600.4	235 502	237 063	0.66	0.92	6 500	52.3

Appendix E

Real plant layouts

This appendix plots the nine real plant layouts of Section 5.3.1, one per page. Each figure draws the transcribed site model—modules grouped into arrays, the site boundary, ground obstacles, and transformer locations—together with the DC graph $\vec{G}^{\text{DC}}$ and AC graph $\vec{G}^{\text{AC}}$ of Chapter 3 that are computed from it. They give the geometric context behind the aggregate properties listed in Table 5.5; the fragmentation of instance P7, in particular, is visible as a scatter of short arrays separated across the site.

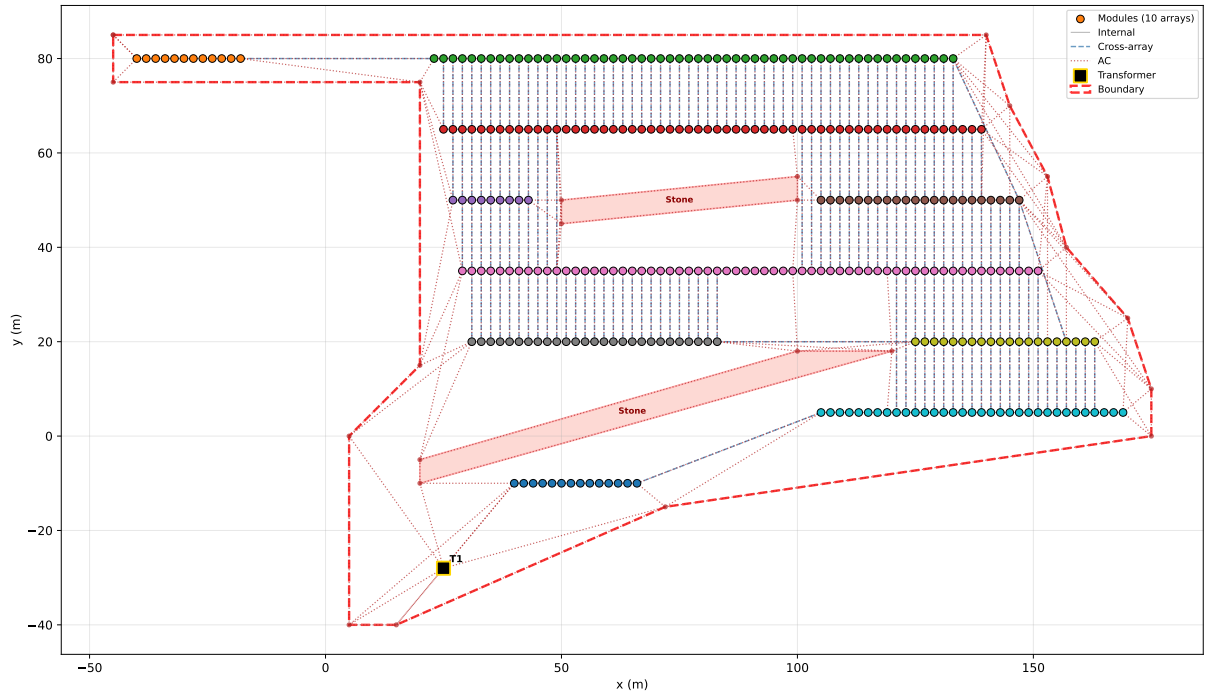


Figure E.1: Real layout of instance P1: 313 modules in 10 arrays served by 1 transformer, and 2 ground obstacles (shaded). Module colour encodes array membership; edges show the DC graph $\vec{G}^{DC}$ (grey intra-array, blue dashed cross-array) and the AC graph $\vec{G}^{AC}$ (red dotted feeders), with the site boundary (red dashed). Properties are tabulated in Table 5.5.

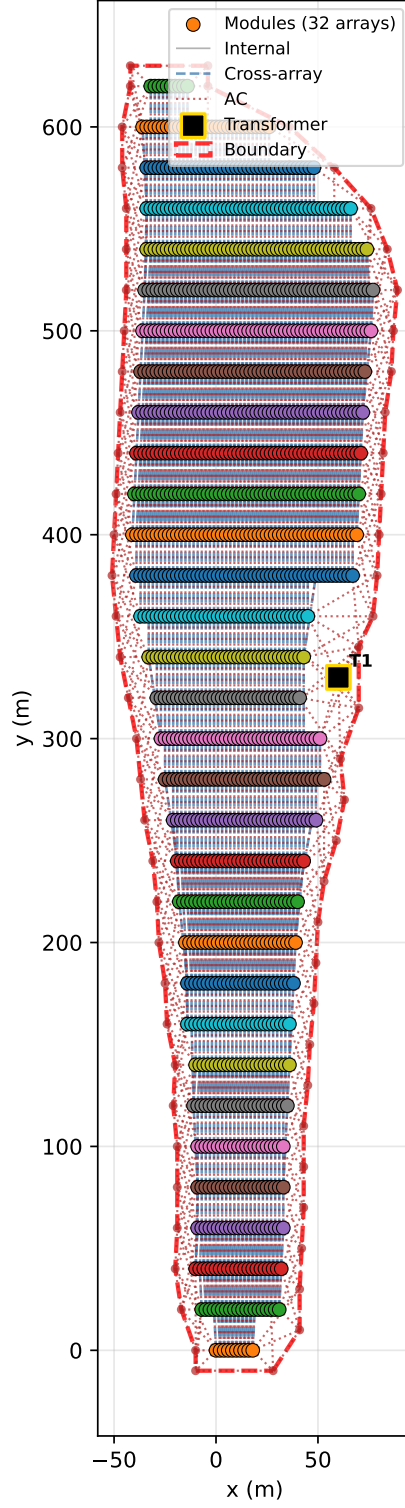


Figure E.2: Real layout of instance p2: 1180 modules in 32 arrays served by 1 transformer. Module colour encodes array membership; edges show the DC graph $\vec{G}^{DC}$ (grey intra-array, blue dashed cross-array) and the AC graph $\vec{G}^{AC}$ (red dotted feeders), with the site boundary (red dashed). Properties are tabulated in Table 5.5.

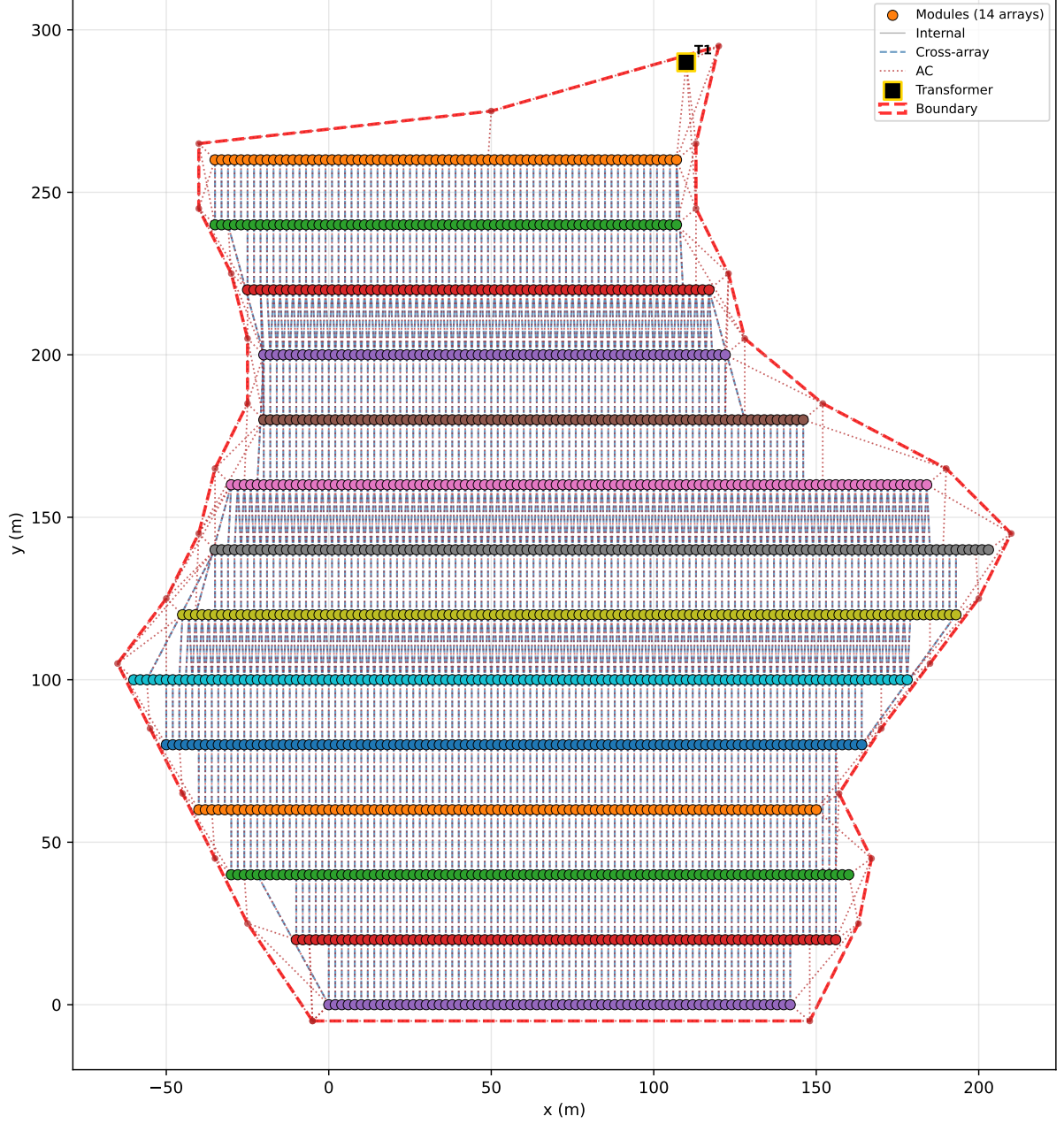


Figure E.3: Real layout of instance P3: 1296 modules in 14 arrays served by 1 transformer. Module colour encodes array membership; edges show the DC graph $\vec{G}^{\text{DC}}$ (grey intra-array, blue dashed cross-array) and the AC graph $\vec{G}^{\text{AC}}$ (red dotted feeders), with the site boundary (red dashed). Properties are tabulated in Table 5.5.

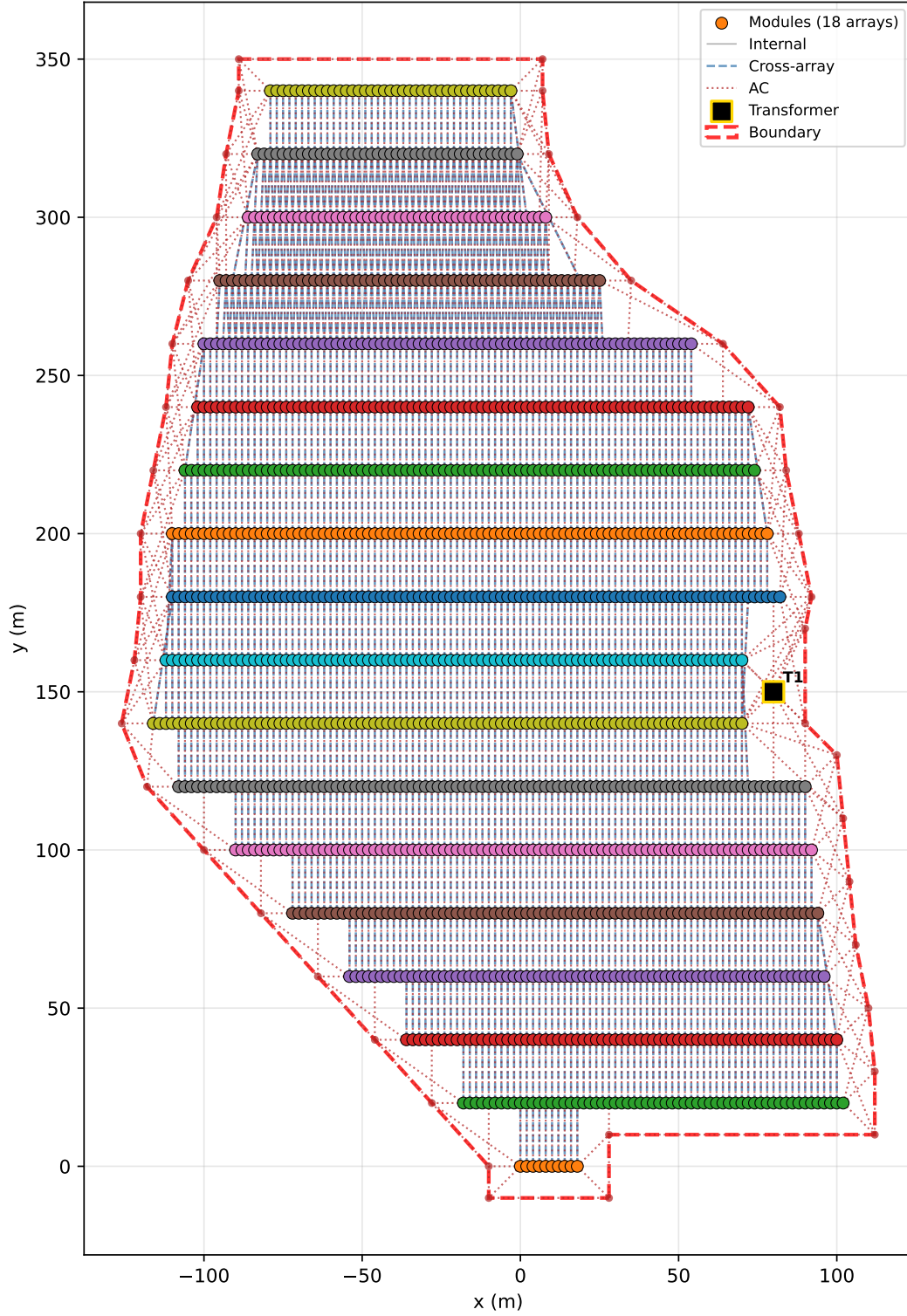


Figure E.4: Real layout of instance P4: 1317 modules in 18 arrays served by 1 transformer. Module colour encodes array membership; edges show the DC graph $\vec{G}^{\text{DC}}$ (grey intra-array, blue dashed cross-array) and the AC graph $\vec{G}^{\text{AC}}$ (red dotted feeders), with the site boundary (red dashed). Properties are tabulated in Table 5.5.

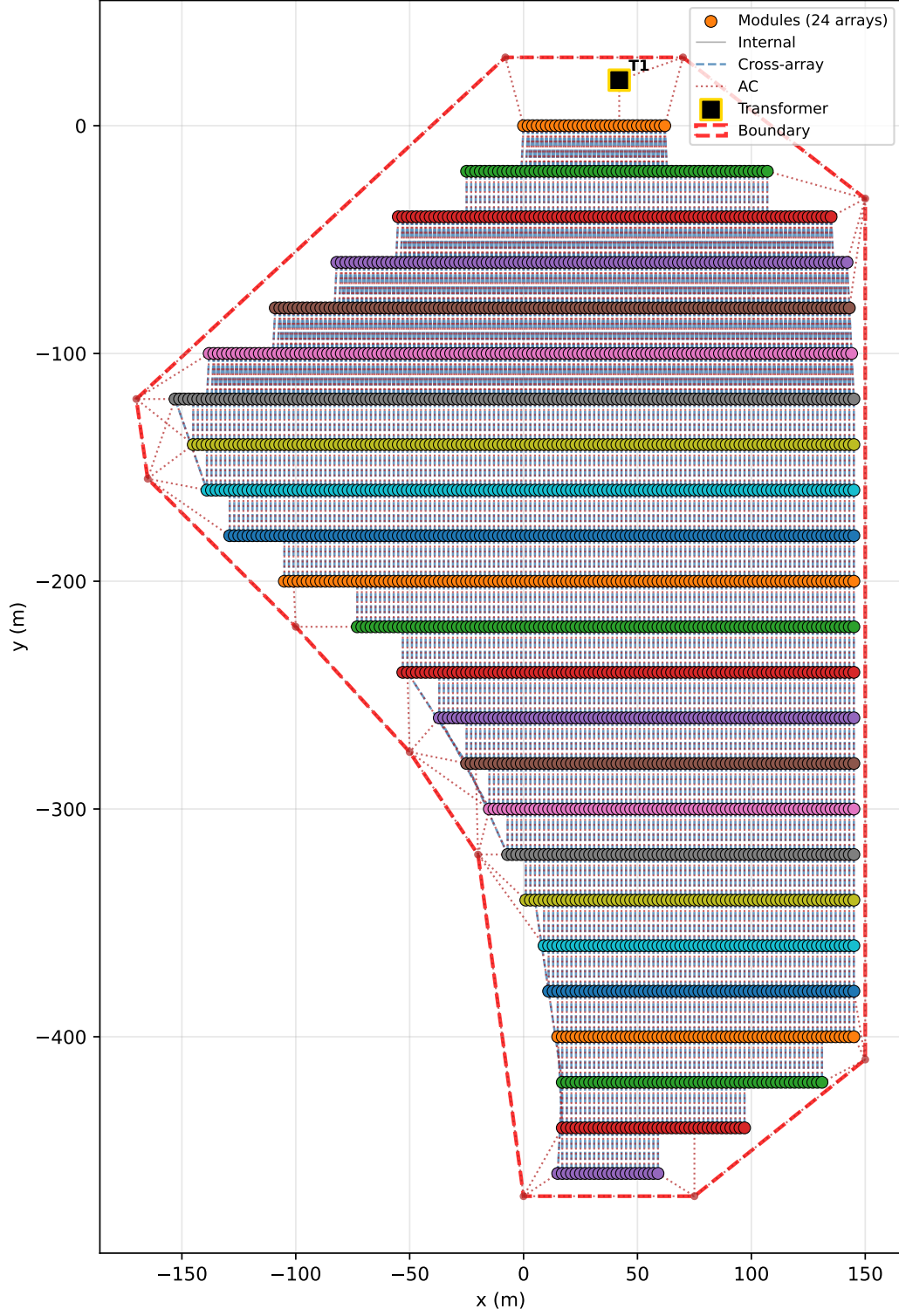


Figure E.5: Real layout of instance p5: 2224 modules in 24 arrays served by 1 transformer. Module colour encodes array membership; edges show the DC graph $\vec{G}^{\text{DC}}$ (grey intra-array, blue dashed cross-array) and the AC graph $\vec{G}^{\text{AC}}$ (red dotted feeders), with the site boundary (red dashed). Properties are tabulated in Table 5.5.

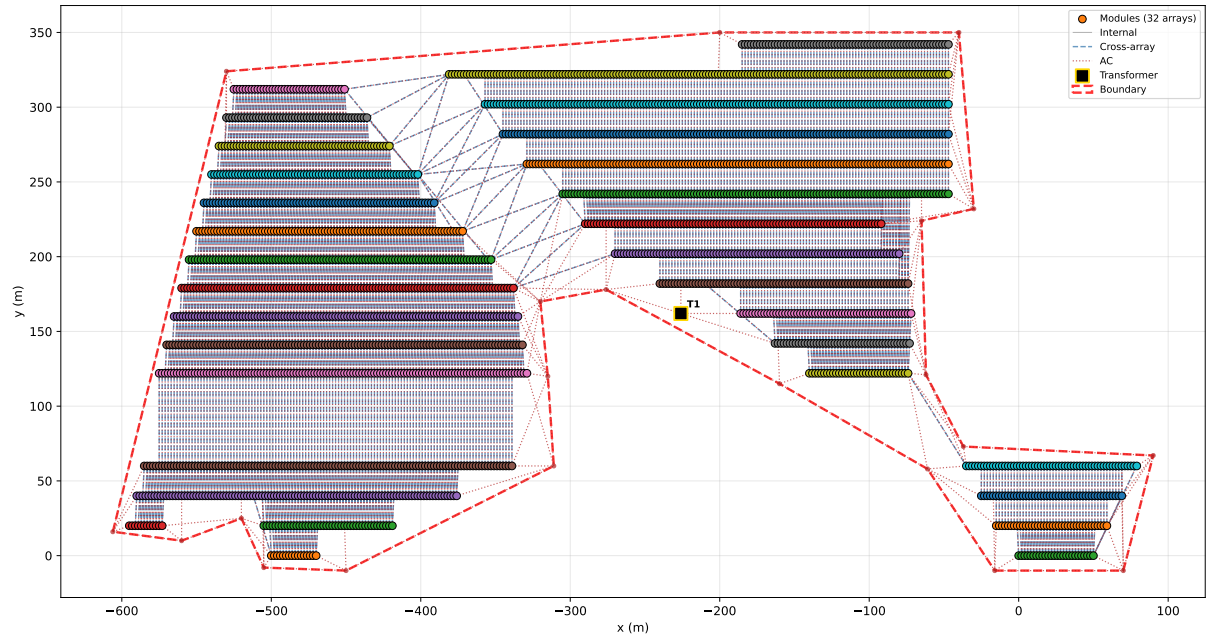


Figure E.6: Real layout of instance P6: 2664 modules in 32 arrays served by 1 transformer. Module colour encodes array membership; edges show the DC graph $\vec{G}^{\text{DC}}$ (grey intra-array, blue dashed cross-array) and the AC graph $\vec{G}^{\text{AC}}$ (red dotted feeders), with the site boundary (red dashed). Properties are tabulated in Table 5.5.

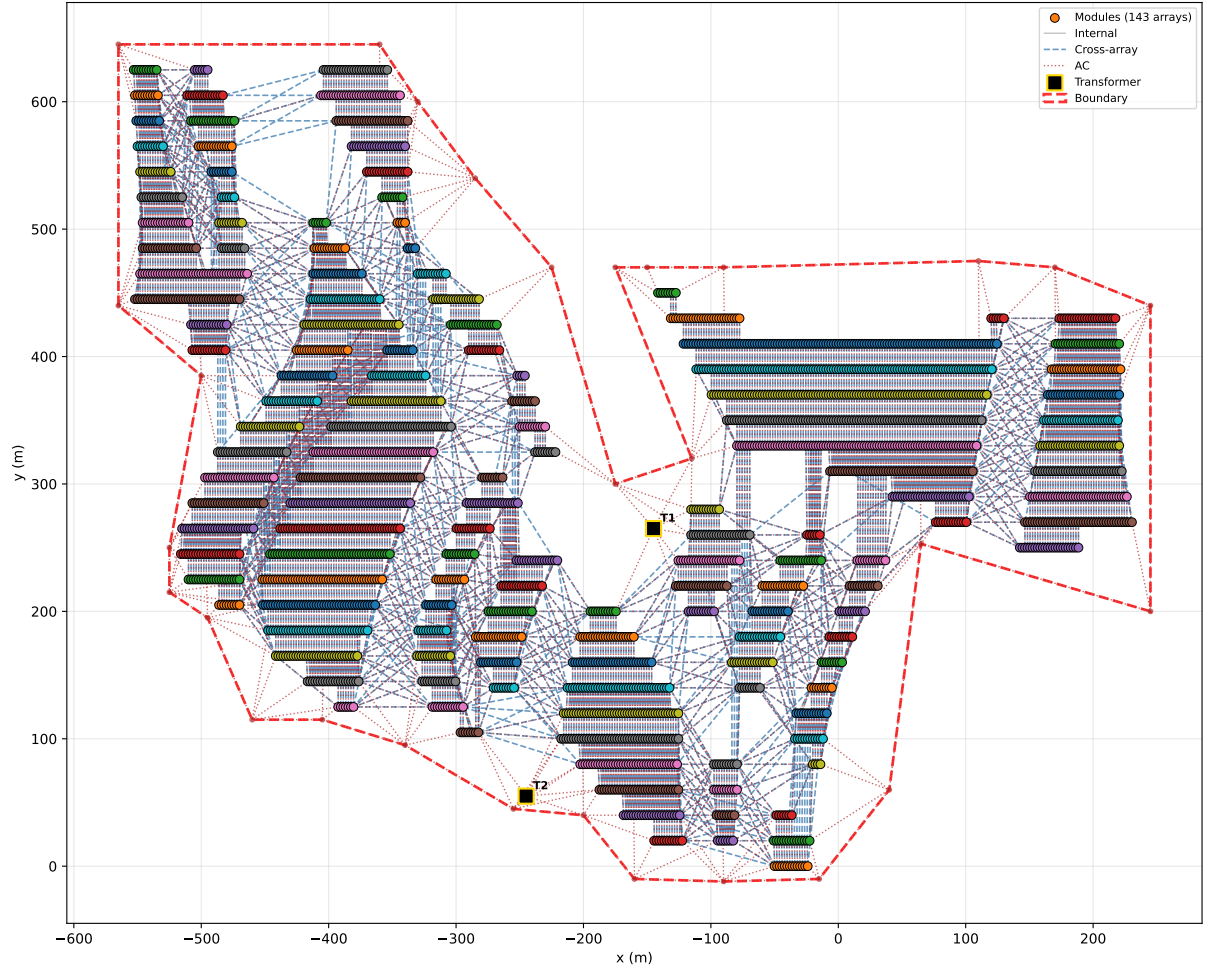


Figure E.7: Real layout of instance P7: 3357 modules in 143 arrays served by 2 transformers. Module colour encodes array membership; edges show the DC graph $\vec{G}^{\text{DC}}$ (grey intra-array, blue dashed cross-array) and the AC graph $\vec{G}^{\text{AC}}$ (red dotted feeders), with the site boundary (red dashed). Properties are tabulated in Table 5.5.

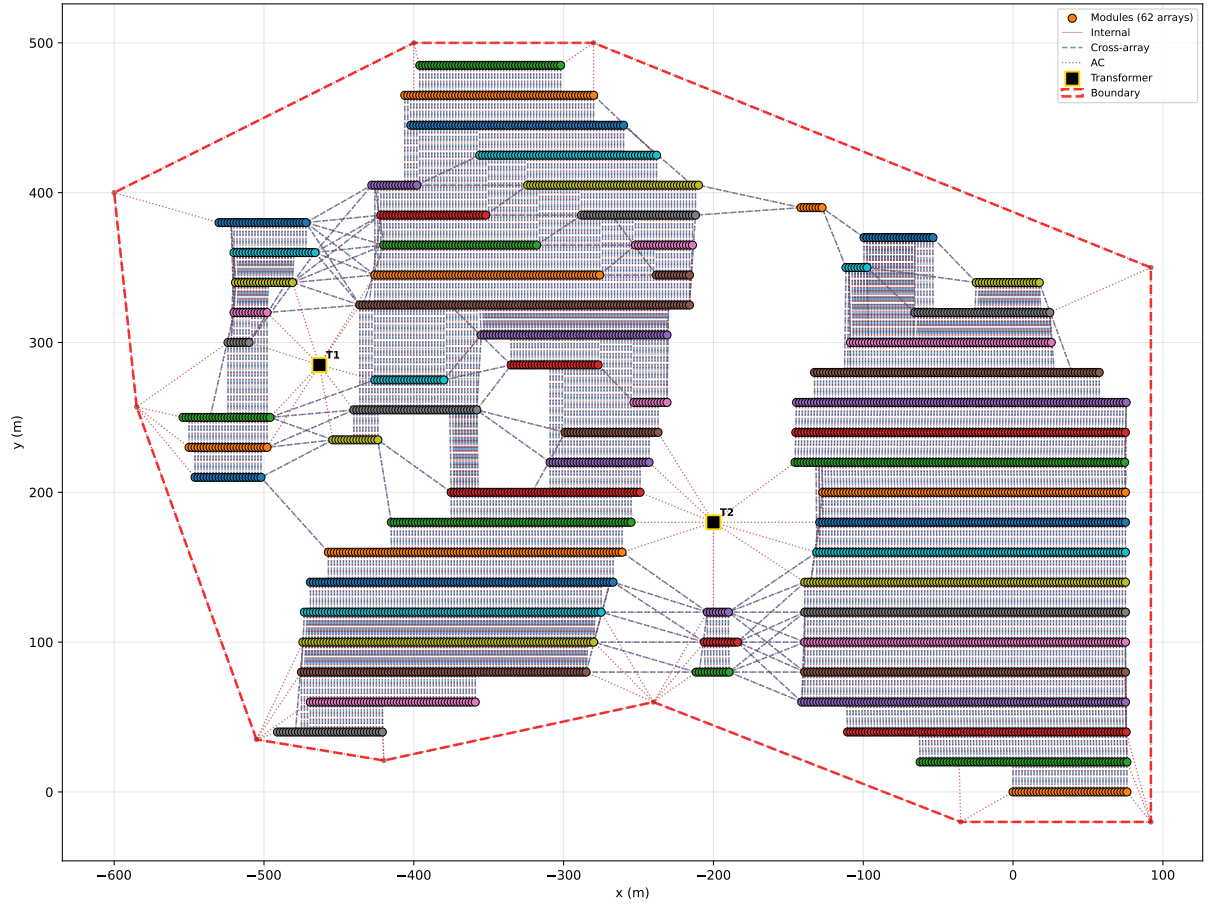


Figure E.8: Real layout of instance p8: 3517 modules in 62 arrays served by 2 transformers. Module colour encodes array membership; edges show the DC graph $\vec{G}^{\text{DC}}$ (grey intra-array, blue dashed cross-array) and the AC graph $\vec{G}^{\text{AC}}$ (red dotted feeders), with the site boundary (red dashed). Properties are tabulated in Table 5.5.

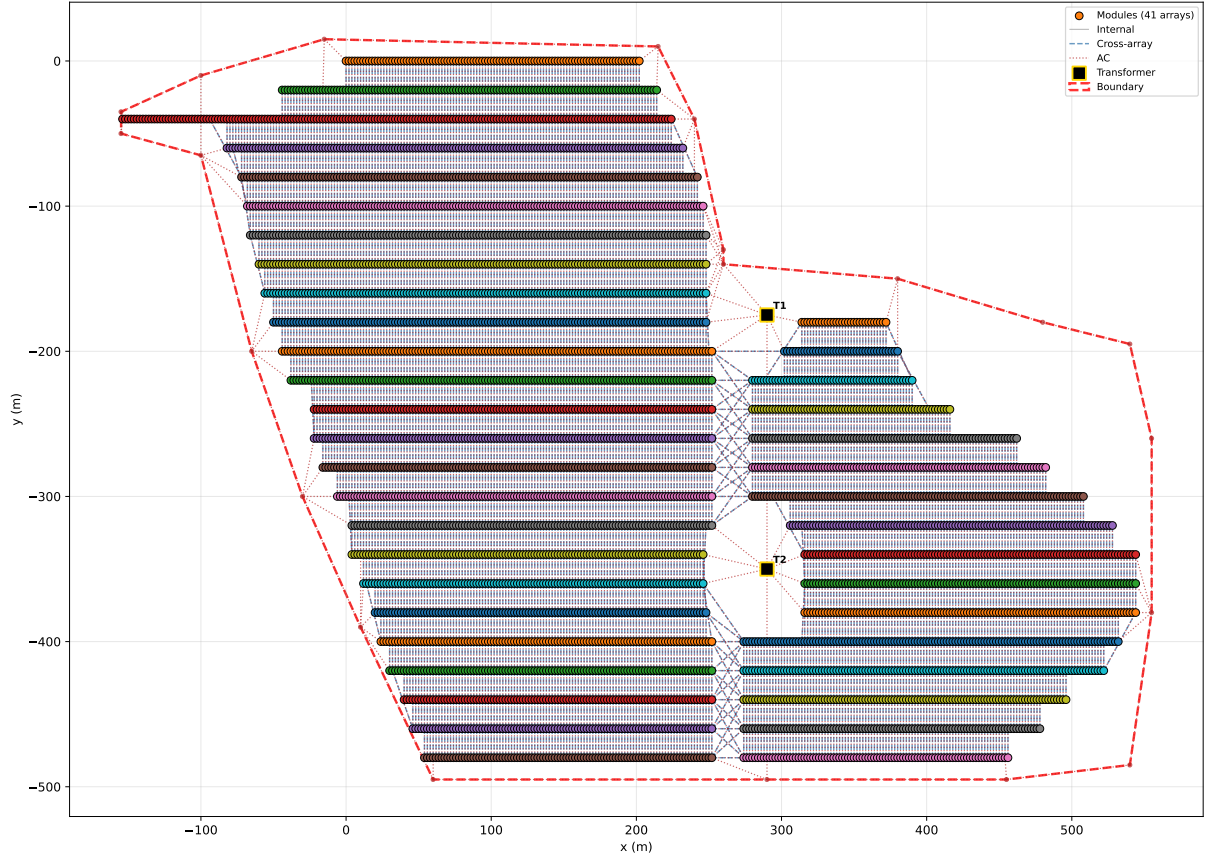


Figure E.9: Real layout of instance P9: 4889 modules in 41 arrays served by 2 transformers. Module colour encodes array membership; edges show the DC graph $\vec{G}^{DC}$ (grey intra-array, blue dashed cross-array) and the AC graph $\vec{G}^{AC}$ (red dotted feeders), with the site boundary (red dashed). Properties are tabulated in Table 5.5.

Appendix F

Staged solver detail

This appendix tabulates the measurements behind Section 6.6: the objective cost and the routed cable lengths and inverter counts of both designs, the per-stage solver telemetry of the staged decomposition, and the layouts of all nine instances under both designs.

F.1 Objective cost breakdown

Table F.1 prices both designs through the five active terms of the Chapter 4 objective; the staged total is lower on every instance. One definition matters when reading the trench column against Table F.2: the objective’s trench term pays for every opened cross-array corridor, whether the corridor carries homerun cable or a PV-string crossing between arrays, while the trench length of Table F.2 counts only the corridors carrying homerun cable. The two differ exactly where a design routes strings across array boundaries—most visibly on P9, where the staged homerun-carrying trench is shorter than the annotation’s yet its trench bill is higher.

The three large-plant trades of Section 6.6.3 read directly off the table. P7 keeps all 14 inverters but digs 238 fewer trench metres in seven fewer trenches; counting also the corridors opened only for cross-array PV-strings, which the objective prices at the same 50 per metre, its trench bill falls by 320 metres, or 16 015 cost units. P8 erases an inverter: the strings it redistributes over fewer collection points travel 4.6 km farther, but at 0.1 per metre that costs 465 units against the 10 000 the inverter returns. On P9 the ledger is tighter. The staged design lays an extra 32.9 km of homerun (+3 288 units), and although its homerun-carrying trench is 34 m shorter, it opens 94 more corridor metres for its cross-array strings (+4 700 units). Even so, the removed inverter repays the whole trade with 2 012 units to spare, and smaller savings in PV-string and feeder cable round the instance’s net to -1.1% .

Inst.	Inverter		DC trench		DC homerun		PV string		AC feeder		Total	
	Ann.	Staged	Ann.	Staged	Ann.	Staged	Ann.	Staged	Ann.	Staged	Ann.	Staged
P1	30,000	30,000	10,739	9,389	324	215	815	701	122	166	42,000	40,472
P2	50,000	50,000	31,017	31,017	907	813	3,292	3,040	335	382	85,552	85,253
P3	50,000	50,000	13,004	13,004	1,022	807	2,834	2,766	289	292	67,149	66,869
P4	50,000	50,000	17,004	15,004	1,198	869	3,082	2,862	222	384	71,506	69,119
P5	90,000	90,000	23,006	22,006	1,934	1,330	5,062	4,742	796	644	120,798	118,723
P6	110,000	110,000	38,875	35,442	2,654	2,175	6,071	5,961	783	1,040	158,383	154,618
P7	140,000	140,000	166,124	150,109	3,778	4,323	11,144	9,649	1,581	1,689	322,627	305,769
P8	140,000	130,000	68,435	66,922	3,478	3,943	8,689	8,138	912	888	221,514	209,890
P9	180,000	170,000	40,100	44,800	5,483	8,771	10,688	10,334	804	510	237,074	234,415

Table F.1: Objective cost of the annotated baseline (*Ann.*) and the staged decomposition (*Staged*) on every instance, split into the five active terms of the Chapter 4 objective; the two overlap-penalty terms are zero for both designs on all instances. Values are rounded to whole cost units, so a total may differ from the sum of its row by one unit. The totals are the exact figures behind Figure 6.2.

F.2 Cable lengths and inverter counts

Table F.2 gives the exact values behind Figure 6.3: the PV-string, DC-homerun, and AC-feeder cable lengths, the DC-trench length, and the inverter count of each design. Two regularities extend beyond the homerun story of Section 6.6.3: the staged trench network is never longer than the annotated one, and the staged PV-string cable is shorter on every instance.

Inst.	DC homerun (m)		AC feeder (m)		PV string (m)		DC trench (m)		Inverters	
	Ann.	Staged	Ann.	Staged	Ann.	Staged	Ann.	Staged	Ann.	Staged
P1	3,239	2,155	122	166	815	701	215	188	3	3
P2	9,070	8,134	335	382	3,292	3,040	620	620	5	5
P3	10,222	8,070	289	262	2,834	2,766	260	260	5	5
P4	11,983	8,691	222	384	3,082	2,862	340	300	5	5
P5	19,335	13,303	776	584	5,062	4,742	460	440	9	9
P6	26,543	21,752	763	975	6,071	5,961	757	709	11	11
P7	37,776	43,234	1,581	1,591	11,144	9,649	3,164	2,926	14	14
P8	34,779	39,425	912	888	8,689	8,138	1,369	1,298	14	13
P9	54,828	87,710	804	510	10,688	10,334	802	768	18	17

Table F.2: Cable lengths and inverter counts of the annotated baseline (*Ann.*) and the staged decomposition (*Staged*) on every instance. Homerun and feeder lengths are the routed circuit lengths behind Figure 6.3; PV-string and trench lengths and the inverter counts locate where the two designs spend inverters and trench.

F.3 Per-stage solver detail

Table F.3 lists, for each instance and stage, the model size, the wall-clock slice, the terminal optimality gap, and the objective in that stage’s own units. The Stage 1 objective prices the aggregated-flow surrogate of Section 6.6.1—in Constraint (OBJ-S1) a trench metre costs 500, an AC metre 1, and a metre of aggregated DC flow 0.0025—and the Stage 2 objective the path-resolution subproblem, which scores Constraint (OBJ-S2) with the real per-metre weights (PV cable 1, trench 50); neither is comparable to the full-system cost. Only the Stage 3 and Stage 4 objectives score the complete Chapter 4 model. The large terminal gaps of Stages 1 and 4 measure the weakness of the respective LP relaxations, diagnosed in Section 5.3, not the distance of the layout from optimality.

Table F.3: Per-stage solver detail for the staged decomposition on every instance. Each stage runs inside its wall-clock slice of the total budget. Variables and constraints give the model size of the stage; the objective is in the stage’s own units and is comparable across stages only for Stages 3 and 4. A dash marks a value Gurobi does not report for the stage.

Inst.	Stage	Vars	Constrs	Time (s)	Gap	Objective
P1	S1	6,284	7,346	2,376	22.8%	124,028
	S2	6,974	4,373	808	21.0%	10,281
	S3	13,190	14,505	1,639	0.2%	40,603
	S4	13,190	14,505	2,367	11.8%	40,472
P2	S1	37,707	40,505	2,376	48.3%	360,697
	S2	63,789	36,882	808	78.9%	34,124
	S3	102,063	93,299	257	0.0%	86,436
	S4	102,063	93,299	3,750	39.0%	85,253
P3	S1	33,605	37,655	2,376	70.1%	180,556
	S2	62,559	36,568	807	75.4%	15,890
	S3	68,575	70,787	1,638	0.4%	66,888
	S4	68,575	70,787	2,363	21.0%	66,869
P4	S1	33,070	37,263	2,376	65.9%	200,549
	S2	61,143	35,913	808	76.1%	17,940
	S3	67,546	70,348	1,641	0.6%	69,267

Inst.	Stage	Vars	Constrs	Time (s)	Gap	Objective
P5	S4	67,546	70,348	2,362	21.9%	69,119
	S1	58,307	65,043	2,376	69.1%	311,068
	S2	194,849	111,019	808	83.0%	26,976
	S3	119,197	122,385	1,639	0.5%	118,814
	S4	119,197	122,385	2,357	20.0%	118,723
P6	S1	76,139	84,287	2,388	66.3%	466,300
	S2	304,855	171,895	805	82.7%	41,631
	S3	153,031	155,175	1,628	0.7%	155,962
	S4	153,031	155,175	2,346	25.0%	154,618
P7	S1	120,884	115,766	2,376	56.6%	1,606,082
	S2	461,494	265,991	807	94.1%	156,127
	S3	374,674	322,987	1,635	1.1%	305,769
	S4	374,674	322,987	2,345	57.1%	305,769
P8	S1	110,847	100,668	2,376	78.0%	781,319
	S2	416,433	236,997	808	–	73,538
	S3	361,291	312,062	1,636	1.1%	209,895
	S4	361,291	312,062	2,336	37.7%	209,890
P9	S1	147,338	133,390	2,376	12.9%	313,450
	S2	729,510	413,754	805	–	192,620
	S3	486,098	423,795	1,635	0.6%	234,415
	S4	486,098	423,795	2,336	27.1%	234,415

F.4 Layouts of all nine instances

Figure F.1 shows every instance under both designs. The visual pattern of Section 6.6.3 holds across the set: each pair fills the same module field, trenches along the same corridors, and differs in terminal orientations and string-to-inverter pairings that the eye does not resolve. The largest instance P9 (4 889 modules) carries the sharpest inverter-for-cable trade: the staged decomposition places 17 inverters where the annotation places 18 and lays 87.7 km of homerun cable against the baseline’s 54.8 km.

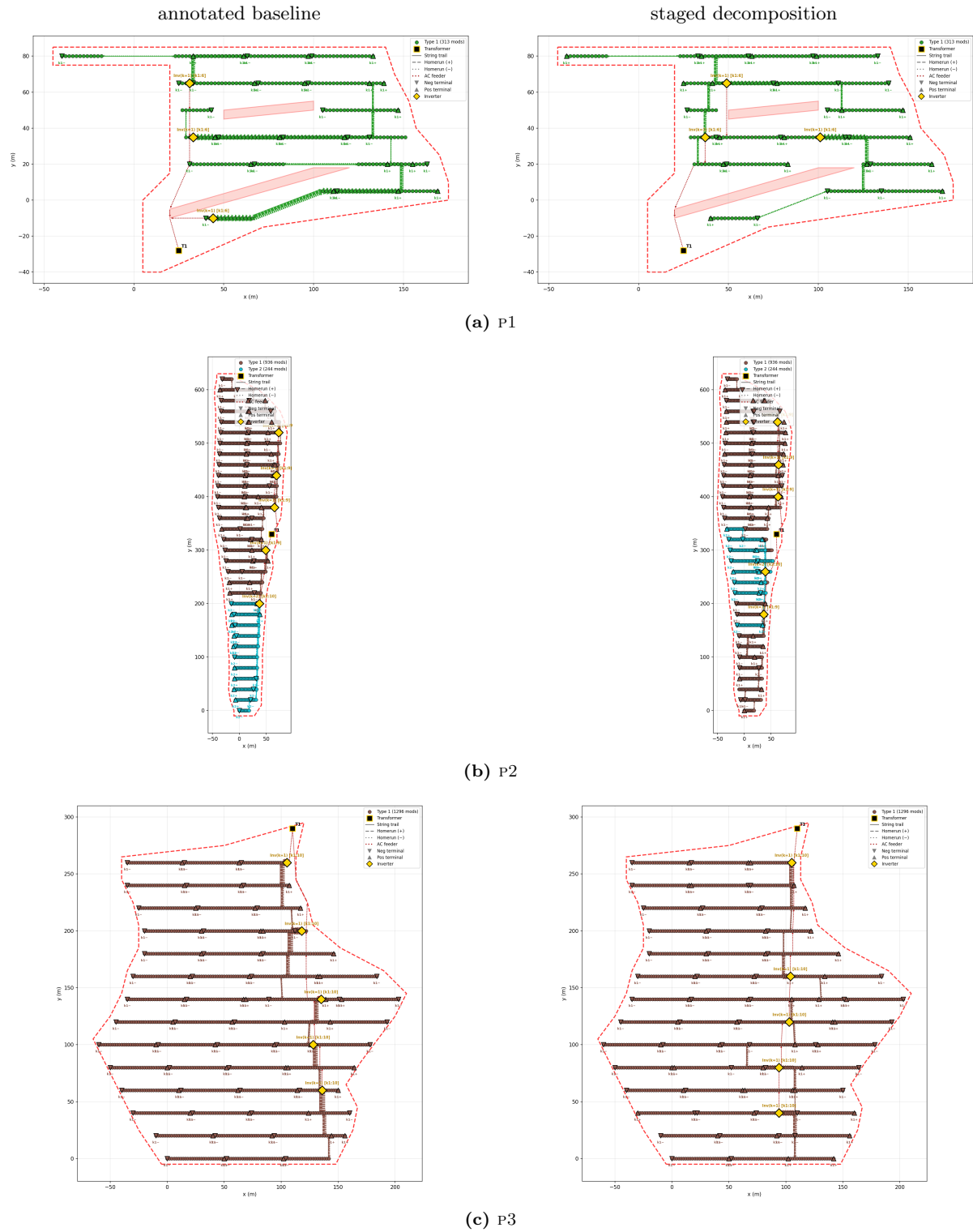
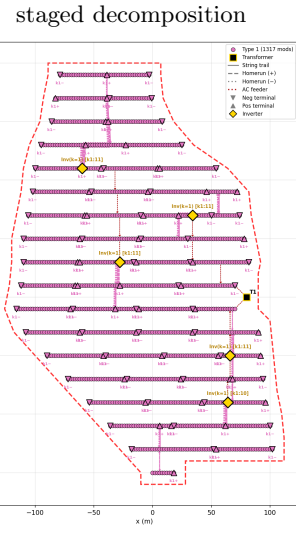
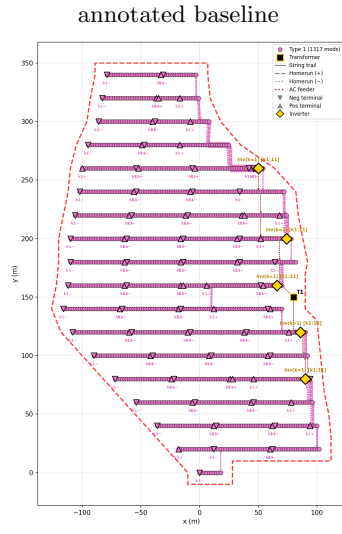
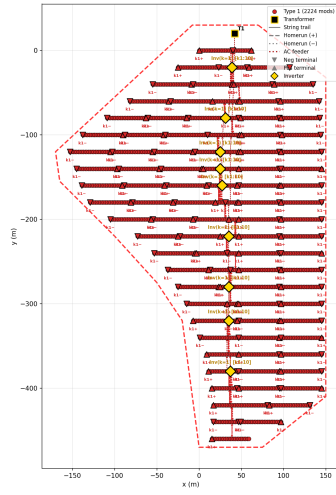
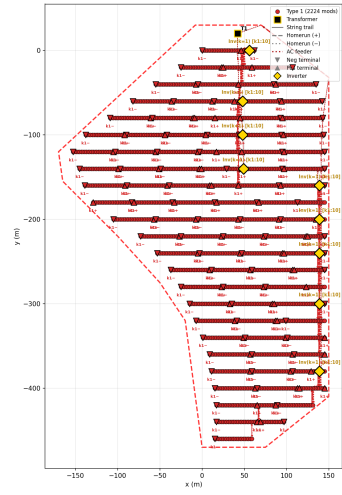


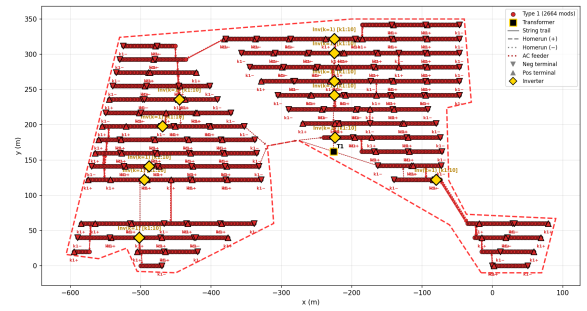
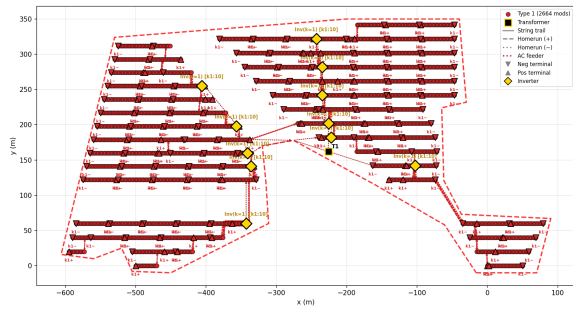
Figure F.1: All nine plants under both designs: the annotated baseline in the left column, the staged decomposition in the right. The paired layouts are visually interchangeable at this scale, and Table F.2 carries the differences: on P1–P6 the staged decomposition delivers the engineer’s inverter and string counts with 10–34% less homerun cable, and on P7–P9 it stretches the homeruns to remove an inverter (P8, P9) or trench (P7). This page: P1–P3.



(d) P4



(e) P5



(f) P6

Figure F.1: All nine plants under both designs (continued): P4–P6.

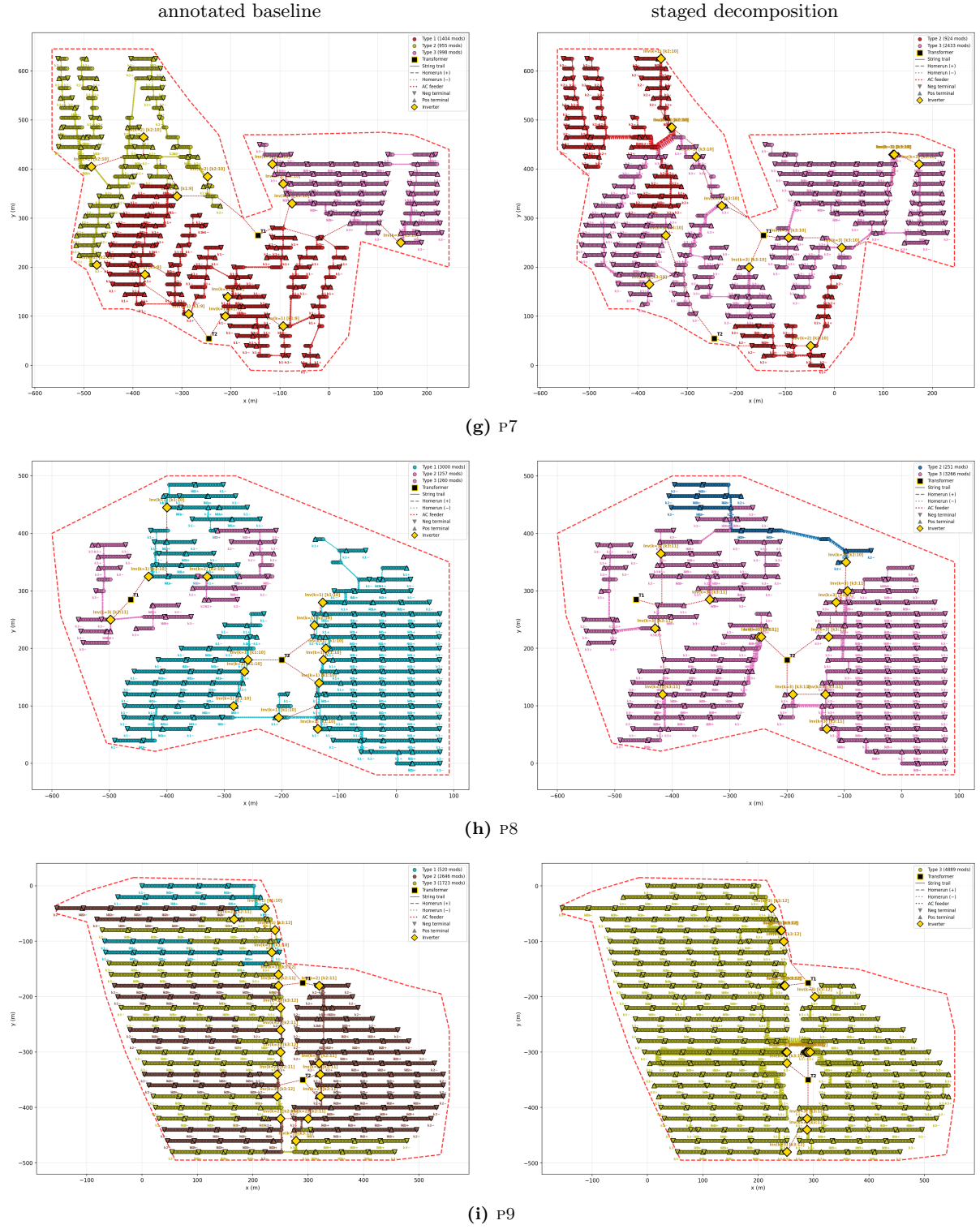


Figure F.1: All nine plants under both designs (continued): P7-P9.