Facility Location with Decision-Dependent Scenario Probabilities: Linked Lifted Cuts and a Scenario-Decomposed Branch-and-Cut

Karyamsetty Martin

November 2026

Abstract

We study two-stage facility location in which the probability of each demand scenario is an affine function of the opening decisions, so that the objective $f^\top y + \sum_s p_s(y)\, Q_s(y)$ is bilinear in the binary first stage y and the recourse values $Q_s(y)$. The standard treatment linearizes the products $y_j\theta_s$ by McCormick envelopes over global bounds on θ_s; on our instances its LP relaxation has a root gap of 6–10%. We propose a *linked lifted-cut master*: the posiform of $p_s(y)$ splits each product into factors $y_jQ_s(y)$ and $(1-y_j)Q_s(y)$, each factor is the value of the recourse LP at a scaled right-hand side, and every vertex of the shared recourse dual polyhedron therefore yields a cut that is linear in (y, Y) with $Y = yy^\top$ lifted. Linking rows tie the factor epigraphs of one scenario together. The relaxation is exact on binary points and has root gaps of 0.07–0.43% on the same instances. Because the linked LP is dual block-angular with closed-form scenario blocks, its value is computed by a stabilized multicut cutting-plane master over (y, Y, θ) whose cuts are valid at every node of a branch-and-bound tree. The resulting branch-and-cut proves optimality on all 25 test instances with 25 to 50 sites and 20 scenarios, in 4 s to 12 min, while a McCormick branch-and-cut sharing the same tree code leaves seven of fifteen unproven after two to four hours and, with node-local recourse bounds, still leaves six of 25 unproven and needs three to four hundred times more nodes; a full MIP solver closes the McCormick gap faster than that tree but is still 1.6–13× slower than the decomposed solver. A node-parallel MPI version of the solver reaches 1.6–3.3× on seven workers, and a sparse-influence variant of the lifting proves instances with 500 sites in under four minutes.

1 Introduction

In the classical two-stage stochastic facility location problem the scenario probabilities are data. In many applications they are not: opening a flagship store shifts demand toward it, a new depot changes which disruption scenarios matter, a hub makes the high-volume scenario more likely. Stochastic programs in which the decision influences the probability measure are called problems with *endogenous uncertainty* of the decision-dependent-probability type [5, 7]. They are hard for a structural reason: the expectation $\sum_s p_s(y)Q_s(y)$ is a sum of products of a function of y and a function of y, and convexity of Q_s is not inherited by the product.

The published computational approaches either hand the bilinear objective to a global solver [7] or linearize the products $y_j\theta_s$ of a binary and a bounded continuous variable with McCormick envelopes, which are exact at binary points [3]. The McCormick relaxation is, however, weak: between a binary point and the next, the product may sit anywhere in a box whose width is the whole range of the recourse value. The branch-and-bound tree then has to do what the relaxation does not.

This paper makes three contributions.

1. A *linked lifted-cut* LP relaxation of the problem that is exact on binary points and whose root gap is one to two orders of magnitude smaller than the McCormick relaxation on the same instances (Section 4).

2. A *scenario decomposition* of that LP in which every scenario block has a closed-form value and subgradient, so the node bound of a branch-and-bound tree is a small stabilized cutting-plane master whose cuts remain valid throughout the tree (Section 5).
3. A branch-and-cut built on it, with a computational study against the McCormick master in the same tree code, with static and with node-local bounds, and inside a full MIP solver; an ablation of the linking rows; sensitivity in the number of scenarios, the strength of the endogeneity and the number of customers, and a $b_s = 0$ variant on which the linking rows are vacuous; a C++ implementation; a node-parallel MPI version; and a sparse-influence variant of the lifting that reaches 500 sites (Sections 6–8).

2 Problem

Sites $i = 1, \dots, n$ with capacity cap_i and fixed cost f_i serve customers $k = 1, \dots, m$. Scenario $s = 1, \dots, S$ has demands $d_s \in \mathbb{R}^m_+$ and unit transport costs C_{ik}; unmet demand costs pen per unit. The first stage is $y \in \{0,1\}^n$. The recourse value is the transportation LP

$$Q_s(y) = \min_{x,u \geq 0} \Big\{ \sum_{ik} C_{ik} x_{ik} + \mathrm{pen} \sum_k u_k \;:\; \sum_i x_{ik} + u_k \geq d_{sk} \;\forall k,\; \sum_k x_{ik} \leq \mathrm{cap}_i y_i \;\forall i \Big\}, \quad (1)$$

whose dual is

$$Q_s(y) = \max_{(\alpha,\beta) \in \mathcal{D}} \alpha^\top d_s - \sum_i \beta_i \mathrm{cap}_i y_i, \quad \mathcal{D} = \{(\alpha,\beta) : \alpha_k - \beta_i \leq C_{ik},\; 0 \leq \alpha \leq \mathrm{pen},\; 0 \leq \beta \leq \mathrm{pen}\}. \quad (2)$$

(Demand satisfaction is written as an inequality; since all costs are positive it is equivalent to the equality form and gives $\alpha \geq 0$, and the unmet-demand variables give $\alpha \leq \mathrm{pen}$. No primal variable imposes $\beta \leq \mathrm{pen}$: it is a valid truncation. Given any dual-feasible (α, β), replacing β_i by $\max(0, \max_k(\alpha_k - C_{ik})) \leq \mathrm{pen}$ keeps feasibility, since $C \geq 0$, and does not lower the objective, whose coefficient $-\mathrm{cap}_i y_i$ of β_i is nonpositive. Hence the truncation changes no value $Q_s(y)$, $y \geq 0$, and makes $\mathcal{D}$ a polytope even where $y_i = 0$ leaves β_i free; $\mathcal{D}$ is the convex hull of its vertices. The rays of the untruncated dual cone, $\beta_i \to \infty$ where $y_i = 0$, are therefore never needed: for every $y \in [0,1]^n$ the value $Q_s(y)$ is attained at a vertex of the truncated $\mathcal{D}$, and every cut below is built from those vertices alone.) Two facts about (2) carry the paper. First, $\mathcal{D}$ does not depend on y, so every vertex (α, β) of $\mathcal{D}$ gives the Benders cut $Q_s(y) \geq \alpha^\top d_s - (\beta \circ \mathrm{cap})^\top y$, valid for every y. Second, write $Q(r, c)$ for the value of (1) with demand r and capacity c; then Q is positively homogeneous, $Q(\lambda r, \lambda c) = \lambda Q(r, c)$ for $\lambda \geq 0$, and the same vertices of $\mathcal{D}$ are optimal.

Remark 1 (scope of the two facts)**.** *Neither fact is specific to transportation. For any second-stage LP $Q(h) = \min\{q^\top x : Wx \geq h,\; x \geq 0\}$ with fixed recourse matrix W and right-hand side $h(y) = h_0 + Ty$ affine in y, the dual feasible set $\{\pi \geq 0 : W^\top \pi \leq q\}$ does not depend on y and Q is positively homogeneous in h, so $y_j Q(h(y)) = Q(y_j h_0 + T(y_j y))$ and $(1 - y_j) Q(h(y)) = Q((1 - y_j) h_0 + T(y - y_j y))$ are values of the same LP at right-hand sides affine in $(y, yy^\top)$, and every dual vertex yields a cut linear in the lifted variables. Everything in Sections 4–5 uses only this and a dual feasible set that is a polytope, or can be replaced by one without changing the value at any right-hand side that occurs. The first is a structural condition: the dual of $Q(h)$ is bounded iff $\{x \geq 0 : Wx > 0\}$ is nonempty (Motzkin's transposition theorem), which complete recourse with a penalised slack on every row guarantees. The transportation LP has slacks on the demand rows only and its capacity duals are unbounded where $y_i = 0$; the truncation $\beta \leq \mathrm{pen}$ is valid there because of the bipartite structure $\alpha_k - \beta_i \leq C_{ik}$ with $C \geq 0$. Another recourse structure without a slack on every row needs a truncation argument of its own, or the feasibility rays carried as cuts.*

The scenario probabilities depend on y through an affine map that is a probability vector on every binary point:

$$p_s(y) = b_s + a_s^\top y, \qquad \sum_s b_s = 1, \quad \sum_s a_s = 0, \quad \mathrm{bpos}_s := b_s + \sum_j \min(a_{sj}, 0) \geq 0. \tag{3}$$

The last condition is equivalent to $p_s(y) \geq 0$ for all $y \in \{0,1\}^n$. The problem is

$$\min_{y \in \{0,1\}^n} \; F(y) = f^\top y + \sum_s p_s(y)\, Q_s(y). \tag{4}$$

With exogenous probabilities ($a_s = 0$) this is a standard Benders decomposition. The products $p_s(y)Q_s(y)$ are the entire difficulty.

Remark 2 (scenario-dependent recourse data)**.** *Nothing below uses that C,* cap *and* pen *are the same in every scenario. With scenario data C^s,* cap^s, pen^s *the dual polyhedron becomes $\mathcal{D}_s$, still independent of y, and every statement and proof goes through with $\mathcal{D}_s$,* cap^s *in place of $\mathcal{D}$,* cap*; the implementation already keeps one vertex list per scenario (Section 4), so it covers this case as written. We keep the shared polyhedron in the notation only because it lets one vertex serve every scenario.*

Instances. All computations use one generator: $m = 15$, $S = 20$, costs $C_{ik} \sim U(1,5)$, demands $d_{sk} \sim U(1.5, 4.5)$ scaled by a factor rising from 0.5 to 1.6 across scenarios, total capacity about 1.8× the mean total demand, pen = 10, and fixed costs calibrated as the recourse saving of each site at the mean demand times $U(0.6, 1.6)$, so that opening a strict subset is optimal. In the "strong" endogeneity used throughout, the low-demand half of the scenarios carries the probability mass at $y = 0$ and each opened site moves a share of up to 90% of that mass to the high-demand scenarios: facilities attract demand. Three seeds per size.

3 The McCormick master and why it is weak

Introduce $\theta_s \geq Q_s(y)$ with Benders cuts and expand $p_s(y)\theta_s = b_s\theta_s + \sum_j a_{sj}\, y_j\theta_s$. With valid bounds $Q_s^{\min} \leq \theta_s \leq Q_s^{\max}$, each product $y_j\theta_s$ is replaced by z_{sj} in its McCormick envelope

$$z_{sj} \geq Q_s^{\min} y_j, \quad z_{sj} \geq \theta_s - Q_s^{\max}(1 - y_j), \quad z_{sj} \leq Q_s^{\max} y_j, \quad z_{sj} \leq \theta_s - Q_s^{\min}(1 - y_j). \tag{5}$$

Because y_j is binary the envelope is exact on integer points [3], so minimizing $f^\top y + \sum_s (b_s\theta_s + \sum_j a_{sj} z_{sj})$ over (5) and the Benders cuts is a correct MILP. We call it MC-B. The only valid bounds available before solving are $Q_s^{\max} = \mathrm{pen} \sum_k d_{sk} = Q_s(\mathbf{0})$ (nothing open) and $Q_s^{\min} = Q_s(\mathbf{1})$ (everything open); both hold for every $y \in [0,1]^n$ because Q_s is nonincreasing in y ($\beta \geq 0$ in (2)). At a fractional y the envelope lets z_{sj} sit anywhere in a box of width $Q_s^{\max} - Q_s^{\min}$, the full range of the recourse, and the LP picks the cheapest corner. Nothing in (5) connects the value of θ_s at a point to the value of the product at that point. On our instances the root gap of MC-B is 6.1–9.7% (Table 2).

4 The linked lifted-cut master

The idea is to give each product $y_jQ_s(y)$ its own exact convex under-estimator rather than relax $y_j\theta_s$ through global bounds. Three observations make this possible.

Posiform. Since $\mathrm{bpos}_s \geq 0$, the probability splits into nonnegative pieces by the sign of a_{sj}:

$$p_s(y) = \mathrm{bpos}_s + \sum_{j:a_{sj}>0} a_{sj}\, y_j + \sum_{j:a_{sj}<0} |a_{sj}|\,(1-y_j) \qquad (y \in \{0,1\}^n), \tag{6}$$

so $p_s(y)Q_s(y)$ is a nonnegative combination of three kinds of factors: $Q_s(y)$, $y_j Q_s(y)$ and $(1-y_j)Q_s(y)$.

Factors are recourse LPs. For $y \in \{0,1\}^n$ let $Y = yy^\top$ and let Y_j denote its j-th row, so $Y_{ji} = y_j y_i$ and $Y_{jj} = y_j$. By homogeneity,

$$y_j\, Q_s(y) = Q\big(y_j d_s,\ \mathrm{cap} \circ Y_j\big), \qquad (1-y_j)\, Q_s(y) = Q\big((1-y_j)d_s,\ \mathrm{cap} \circ (y - Y_j)\big). \tag{7}$$

Both are values of the same dual polyhedron $\mathcal{D}$ at a different right-hand side. Hence every vertex (α, β) of $\mathcal{D}$, with $g := \beta \circ \mathrm{cap}$, yields cuts that are *linear in* (y, Y):

$$e_s^0 \geq \alpha^\top d_s - g^\top y, \tag{8}$$

$$e_{sj}^+ \geq \alpha^\top d_s\, y_j - g^\top Y_j, \tag{9}$$

$$e_{sj}^- \geq \alpha^\top d_s\,(1-y_j) - g^\top (y - Y_j). \tag{10}$$

Remark 3 (conditional configurations)**.** *At a master point $(\hat{y}, \hat{Y})$ with $0 < \hat{y}_j < 1$, homogeneity gives $Q(\hat{y}_j d_s, \mathrm{cap}\circ\hat{Y}_j) = \hat{y}_j\, Q_s(\hat{Y}_j/\hat{y}_j)$ and $Q((1-\hat{y}_j)d_s, \mathrm{cap}\circ(\hat{y}-\hat{Y}_j)) = (1-\hat{y}_j)\, Q_s\big((\hat{y}-\hat{Y}_j)/(1-\hat{y}_j)\big)$, and the McCormick rows $0 \leq Y_{jl} \leq \min(y_j, y_l)$, $y_l - Y_{jl} \leq 1 - y_j$ put both quotients in $[0,1]^n$. The factor cuts are therefore ordinary Benders cuts evaluated at the conditional configurations $\hat{Y}_j/\hat{y}_j$ and $(\hat{y}-\hat{Y}_j)/(1-\hat{y}_j)$, the points the master believes in given $y_j = 1$ and given $y_j = 0$, exactly the points of Balas's lift-and-project and of level-1 RLT. This is why the vertices separated at the factor points are so effective and why the root pool is nearly complete for the tree (Section 6).*

A vertex found for one factor of one scenario is a valid cut for every factor of every scenario. The implementation keeps one list of vertices per scenario, $\{(\alpha_k, \beta_k)\}_{k \leq K_s}$, each list evaluated against its own d_s; a vertex separated for scenario s is added to the list of s only, which is a choice of economy rather than of validity (adding it to every list is sound and was not needed).

Linking. For every j, $y_j Q_s(y) + (1-y_j)Q_s(y) = Q_s(y)$. The master carries this as the rows

$$e_s^0 = e_{sj}^+ + e_{sj}^- \qquad \forall s,\ \forall j. \tag{11}$$

Without them each factor epigraph is bounded only by its own cuts and the three epigraphs of a scenario can sit at inconsistent levels; with them the relaxation tightens substantially (we call the model without (11) *unlinked*; Section 7 reports its root gap as an ablation). Proposition 3 below makes precise how much they tighten: the effect of the linking rows on the value of scenario s is bpos_s times the amount by which they lift e_s^0, so a scenario with $\mathrm{bpos}_s = 0$ gains nothing from them.

The master LP. With pair variables $Y_{jl} = Y_{lj}$, $j < l$, and $Y_{jj} := y_j$, the linked lifted-cut master C2L is

$$\begin{aligned} \min\ & f^\top y + \sum_s \Big[\mathrm{bpos}_s\, e_s^0 + \sum_{a_{sj}>0} a_{sj}\, e_{sj}^+ + \sum_{a_{sj}<0} |a_{sj}|\, e_{sj}^-\Big] \\ \text{s.t. } & Y_{jl} \leq y_j, \quad Y_{jl} \leq y_l, \quad Y_{jl} \geq y_j + y_l - 1, \quad Y_{jl} \geq 0 \qquad (j < l), \\ & \text{(8)–(10) for every vertex in the pool}, \quad (11), \quad y \in [0,1]^n,\ e \geq 0. \end{aligned} \tag{12}$$

Proposition 1 (validity and exactness)**.** *Let $v(y)$ be the value of* (12) *with y fixed. Then $v(y) \le F(y)$ for every $y \in [0,1]^n$, and $v(\hat{y}) = F(\hat{y})$ for every $\hat{y} \in \{0,1\}^n$ at which the pool contains, for each s, a vertex optimal in* (2) *at $\hat{y}$.*

Proof. Validity: fix $y \in [0,1]^n$ and set $Y_{jl} = y_j y_l$, which satisfies the McCormick rows for every y in the box. Set $e^0_s = Q_s(y)$, $e^+_{sj} = y_j Q_s(y)$ and $e^-_{sj} = (1-y_j)Q_s(y)$. By positive homogeneity, $y_j Q_s(y) = Q(y_j d_s,\ y_j \operatorname{cap} \circ y) = Q(y_j d_s,\ \operatorname{cap} \circ Y_j)$ and likewise for $(1-y_j)Q_s(y)$, so each of (8)–(10) is the Benders cut of the corresponding LP at its own optimum and holds; the linking rows (11) hold with equality. The objective at this point is $F(y)$ by (6) (which is an identity for every $y \in [0,1]^n$, not only binary ones, because $\sum_j \min(a_{sj}, 0)$ and the two sums recombine to $b_s + a_s^\top y$). Hence $v(y) \le F(y)$. Exactness: at $\hat{y}$ with the optimal vertex (α^*, β^*) in the pool, cut (8) forces $e^0_s \ge Q_s(\hat{y})$. If $\hat{y}_j = 1$ then $Y_j = \hat{y}$ and (9) at (α^*, β^*) forces $e^+_{sj} \ge Q_s(\hat{y})$; if $\hat{y}_j = 0$ then $\hat{y} - Y_j = \hat{y}$ and (10) forces $e^-_{sj} \ge Q_s(\hat{y})$. These lower bounds are attainable together with (11) (set the other factor of each pair to zero), every objective coefficient is nonnegative, so the minimum is attained at them and equals $F(\hat{y})$ by (6). □

Remark 4. (9)–(10) *are what one obtains by multiplying the dual constraint system by y_j and $(1-y_j)$ and linearizing, i.e. the mixed-binary reformulation–linearization of Adams and Sherali [1] applied on the dual side of the recourse. The dual side matters: the polyhedron $\mathcal{D}$ is shared, so the lifted model is generated by cuts from one pool instead of carrying $S(1+2n)$ copies of the primal recourse, and, as the next section shows, its scenario blocks have closed-form values.*

Dominance over the McCormick master. Let C2R denote (12) without the linking rows (11) (the *unlinked* model). Write $v_{\mathrm{MC}}(y)$, $v_{\mathrm{C2R}}(y)$ and $v_{\mathrm{C2L}}(y)$ for the values of the three masters with y fixed, all three built from the same set of dual vertices.

Proposition 2 (dominance)**.** *If the vertex set contains the zero vertex and a vertex optimal in* (2) *at $y = \mathbf{1}$, then for every $y \in [0,1]^n$*

$$v_{MC}(y) \ \le\ v_{C2R}(y) \ \le\ v_{C2L}(y) \ \le\ F(y),$$

and the same chain holds for the LP relaxations (minima over $y \in [0,1]^n$).

Proof. The last inequality is Proposition 1; the middle one holds because C2L adds constraints to C2R. For the first, fix y and let (Y, e) be optimal for C2R at y; by Proposition 3 without the linking step, $e^0_s = L_0$, $e^+_{sj} = L^+_j$, $e^-_{sj} = L^-_j$ with the levels (13) at (y, Y). Let (α^1, β^1) be the vertex optimal at $\mathbf{1}$, so $\bar{d}^1 - g^{1\top}\mathbf{1} = Q^{\min}_s$. Define a point of MC-B: $\theta_s = L_0$, and for each j with $a_{sj} > 0$, $z_{sj} = \max\{Q^{\min}_s y_j,\ \theta_s - Q^{\max}_s(1-y_j)\}$; for $a_{sj} < 0$, $z_{sj} = \min\{Q^{\max}_s y_j,\ \theta_s - Q^{\min}_s(1-y_j)\}$.

Feasibility. θ_s satisfies every Benders cut since L_0 is their maximum. $\theta_s \ge \bar{d}^1 - g^{1\top} y \ge \bar{d}^1 - g^{1\top}\mathbf{1} = Q^{\min}_s$ because $g^1 \ge 0$ and $y \le \mathbf{1}$; $\theta_s \le \max_k \bar{d}_k \le Q_s(\mathbf{0}) = Q^{\max}_s$ by weak duality at $y = \mathbf{0}$. With $\theta_s \in [Q^{\min}_s, Q^{\max}_s]$ the envelope (5) is nonempty and z_{sj} lies in it by construction.

Objective. Using $b_s = \mathrm{bpos}_s + \sum_{a_{sj}<0} |a_{sj}|$, the block objective of MC-B at this point is $\mathrm{bpos}_s L_0 + \sum_{a_{sj}>0} a_{sj} z_{sj} + \sum_{a_{sj}<0} |a_{sj}|(L_0 - z_{sj})$, so it suffices to show $z_{sj} \le L^+_j$ when $a_{sj} > 0$ and $L_0 - z_{sj} \le L^-_j$ when $a_{sj} < 0$. Let k be the pool index attaining L_0 (the zero vertex when $L_0 = 0$, so that $\bar{d}_k = 0$ and $g_k = 0$) and recall the McCormick rows give $0 \le Y_{jl} \le \min(y_j, y_l)$ and $y_l - Y_{jl} \le 1 - y_j$.

- $L^+_j \ge \bar{d}^1 y_j - g^{1\top} Y_j \ge y_j(\bar{d}^1 - g^{1\top}\mathbf{1}) = Q^{\min}_s y_j$.
- $L_0 - L^+_j \le (\bar{d}_k - g_k^\top y) - (\bar{d}_k y_j - g_k^\top Y_j) = \bar{d}_k(1-y_j) - g_k^\top(y - Y_j) \le \bar{d}_k(1-y_j) \le Q^{\max}_s(1-y_j)$.
- $L^-_j \ge \bar{d}^1(1-y_j) - g^{1\top}(y - Y_j) \ge (1-y_j)(\bar{d}^1 - g^{1\top}\mathbf{1}) = Q^{\min}_s(1-y_j)$.
- $L_0 - L^-_j \le (\bar{d}_k - g_k^\top y) - (\bar{d}_k(1-y_j) - g_k^\top(y - Y_j)) = \bar{d}_k y_j - g_k^\top Y_j \le Q^{\max}_s y_j$.

The first two give $z_{sj} \le L^+_j$ for $a_{sj} > 0$; the last two give $z_{sj} \ge L_0 - L^-_j$ for $a_{sj} < 0$. Hence the MC-B objective at the constructed point is at most $v_{\mathrm{C2R}}(y)$, and $v_{\mathrm{MC}}(y)$, a minimum, is smaller still. □

The proof shows where the strength comes from: the lifted cuts bound each product $y_j Q_s(y)$ by the recourse value at a scaled right-hand side, which is at least as tight as any corner of the envelope (5) at every point and strictly tighter whenever the active dual vertex of the factor differs from the one active at $\mathbf{0}$ or $\mathbf{1}$. Section 7 quantifies the two steps: on our instances the unlinked model already reduces the root gap from 6–10% to 0.26–1.14%, and the linking rows reduce it a further 1.7–6×.

Sparse influence. Let $\mathcal{I} = \{j : a_{sj} \neq 0 \text{ for some } s\}$ be the influencing sites. A factor $y_j Q_s(y)$ or $(1 - y_j)Q_s(y)$ with $j \notin \mathcal{I}$ has zero weight in the objective of (12) and enters only through its linking row (11), which raises e^0_s to at least $e^+_{sj} + e^-_{sj}$. The *sparse* master drops the factors of $j \notin \mathcal{I}$ altogether: cuts (9)–(10) and rows (11) are kept for $j \in \mathcal{I}$ only. It is a relaxation of (12), hence valid, and Proposition 1 still holds because its exactness argument uses the e^0 cuts and the factors of influencing sites only. Its pair columns are $\{(j, l) : j \in \mathcal{I} \text{ or } l \in \mathcal{I}\}$, that is $|\mathcal{I}|\, n - |\mathcal{I}|(|\mathcal{I}| + 1)/2$ instead of $n(n-1)/2$, and the scenario blocks of Section 5 evaluate only the rows Y_j, $j \in \mathcal{I}$, of the lifted matrix. When every site influences demand the two masters coincide.

On our instances the root gap of (12) is 0.07–0.43% against 6.1–9.7% for MC-B (Table 2). The price is size: $n(n-1)/2$ pair columns and cut rows that are dense over them (about 720 nonzeros per row at $n = 50$). Re-solving the monolithic LP at every node is already slow at 35–40 sites; the decomposition removes that cost.

5 Scenario decomposition of the linked master

Fix the coupling variables (y, Y). Scenario s owns the private variables (e^0_s, e^+_s, e^-_s), its cuts and its linking rows, and nothing else, so (12) is dual block-angular. With a pool of K vertices for scenario s, write $\bar{d}_k = \alpha_k^\top d_s$ and $g_k = \beta_k \circ \mathrm{cap}$. The tightest epigraph levels are

$$\begin{aligned} L_0 &= \max\Big(0,\ \max_k\ \bar{d}_k - g_k^\top y\Big), \\ L_j^+ &= \max\Big(0,\ \max_k\ \bar{d}_k\, y_j - g_k^\top Y_j\Big), \\ L_j^- &= \max\Big(0,\ \max_k\ \bar{d}_k\,(1 - y_j) - g_k^\top (y - Y_j)\Big). \end{aligned} \tag{13}$$

The 0 in each maximum is the cut of the zero vertex $(\alpha, \beta) = (0, 0) \in \mathcal{D}$, feasible for every instance and kept in every list, so each level is attained by a vertex of the pool.

Proposition 3 (closed-form block)**.** *For fixed (y, Y) the value of the block of scenario s in* (12) *is*

$$\varphi_s(y, Y) = \mathrm{bpos}_s\, e^{0*}_s + \sum_{a_{sj} > 0} a_{sj}\, L_j^+ + \sum_{a_{sj} < 0} |a_{sj}|\, L_j^-, \qquad e^{0*}_s = \max\Big(L_0,\ \max_j\ L_j^+ + L_j^-\Big). \tag{14}$$

Proof. All objective coefficients of the block are nonnegative, so each variable is pushed to its lower bound subject to (11). Row j of (11) is feasible with $e^+_{sj} \geq L_j^+$, $e^-_{sj} \geq L_j^-$ iff $e^0_s \geq L_j^+ + L_j^-$, which gives e^{0*}_s. Given $e^0_s = e^{0*}_s$: if $a_{sj} > 0$ set $e^+_{sj} = L_j^+$ and $e^-_{sj} = e^{0*}_s - L_j^+ \geq L_j^-$, which costs $a_{sj} L_j^+$ and is the least possible; if $a_{sj} < 0$ the mirror choice costs $|a_{sj}| L_j^-$; if $a_{sj} = 0$ the row is free. □

Remark 5 (what the linking rows buy)**.** *For a pair (s, j) at most one of e^+_{sj}, e^-_{sj} carries a nonzero objective coefficient; the other appears only in its own cuts and in row j of* (11)*. Eliminating it, $e^-_{sj} = e^0_s - e^+_{sj} \geq L_j^-$ when $a_{sj} > 0$, shows that the linking rows do nothing but impose $e^0_s \geq \max_j (L_j^+ + L_j^-)$: the active factor epigraphs stay at their own levels, and the block*

value differs from the unlinked one, $\mathrm{bpos}_s L_0 + \sum a_{sj} L_j^+ + \sum |a_{sj}| L_j^-$*, exactly by* $\mathrm{bpos}_s\,(e_s^{0*} - L_0)$*. The posiform objective of* (12) *is what one gets by substituting* $e_{sj}^+ = e_s^0 - e_{sj}^-$ *for* $a_{sj} < 0$ *into* $b_s e_s^0 + \sum_j a_{sj} e_{sj}^+$*, and* bpos_s *is the weight that remains on* e_s^0 *after that substitution. Hence the tightening from C2R to C2L is carried entirely by the residual base mass: a scenario with* $\mathrm{bpos}_s = 0$*, which happens when every unit of its base probability can be moved away by closing sites, is bounded as in the unlinked model. In the generator of Section 2 every scenario has* $\mathrm{bpos}_s \approx 0.009$*: a tenth of each low-demand scenario's base mass* 0.091 *is immovable, and each high-demand scenario keeps its whole base mass* 0.009 *since nothing is moved away from it. The weight on the linking is therefore small; that it still removes two thirds of the unlinked gap (Section 7) is because* $e_s^{0*} - L_0$ *is large at fractional points, where the factor levels* $L_j^+ + L_j^-$ *sit well above* L_0*.*

Each level in (13) is a maximum of affine functions of (y, Y), so φ_s is convex piecewise-linear in (y, Y), and a subgradient is read off the active pieces (envelope theorem). Let k_0, k_j^+, k_j^- be the indices attaining the maxima in (13) (a level equal to 0 has gradient 0), and let j^* attain $\max_j L_j^+ + L_j^-$. Ties, among the j or between L_0 and $L_{j^*}^+ + L_{j^*}^-$, are exactly the points where φ_s is not differentiable; any choice of the active piece gives an element of the subdifferential of a pointwise maximum of convex functions, and the master cut (18) is valid for any element of it. With $Y_{jj} = y_j$ and Y_{jl} stored once for $j < l$, the gradients of the three levels are, writing ∂_{y_l} and $\partial_{Y_{jl}}$ for the coordinates,

$$\begin{aligned}
&\nabla L_0 : \partial_{y_l} = -g_{k_0,l} \quad (\text{all } l);\\
&\nabla L_j^+ : \partial_{y_j} = \bar{d}_{k_j^+} - g_{k_j^+,j}, \qquad \partial_{Y_{jl}} = -g_{k_j^+,l} \quad (l \neq j);\\
&\nabla L_j^- : \partial_{y_j} = -\bar{d}_{k_j^-}, \qquad \partial_{y_l} = -g_{k_j^-,l} \quad (l \neq j), \qquad \partial_{Y_{jl}} = +g_{k_j^-,l} \quad (l \neq j),
\end{aligned} \tag{15}$$

where $\partial_{Y_{jl}}$ is the derivative with respect to the stored pair $Y_{\min(j,l)\,\max(j,l)}$, contributions of rows j and l to the same stored pair add, and the diagonal $Y_{jj} = y_j$ has been folded into ∂_{y_j}: in L_j^+ the term $-g_{k,j} Y_{jj}$ contributes $-g_{k,j}$ to ∂_{y_j}, in L_j^- the term $-g_{k,j}(y_j - Y_{jj})$ vanishes. The subgradient of φ_s is then

$$\gamma_s = \mathrm{bpos}_s\, \nabla e_s^{0*} + \sum_{a_{sj}>0} a_{sj} \nabla L_j^+ + \sum_{a_{sj}<0} |a_{sj}| \nabla L_j^-, \qquad \nabla e_s^{0*} = \begin{cases} \nabla L_0 & \text{if } e_s^{0*} = L_0,\\ \nabla L_{j^*}^+ + \nabla L_{j^*}^- & \text{otherwise.} \end{cases} \tag{16}$$

Evaluating all S blocks costs one matrix product $G_s Y^\top$ per scenario and nothing else; no LP is solved.

Node bound. The value of (12) at a node with box $[\ell, u]$ on y is

$$\min \Big\{ f^\top y + \sum_s \varphi_s(y, Y) \;:\; \ell \le y \le u,\ (y, Y) \text{ in the McCormick set} \Big\}, \tag{17}$$

a convex problem which we solve as a multicut cutting-plane (Benders) master over (y, Y, θ), $\theta \in \mathbb{R}^S$, with the McCormick rows and the cuts

$$\theta_s \;\ge\; \varphi_s(\hat{y}, \hat{Y}) + \gamma_s^\top \big((y, Y) - (\hat{y}, \hat{Y}) \big), \qquad \gamma_s \text{ from (16) at } (\hat{y}, \hat{Y}). \tag{18}$$

Its value is a lower bound on (17) at every iteration; the best $f^\top z + \sum_s \varphi_s(z)$ seen is an upper bound; they meet at the optimum. When the loop stops at tolerance 10^{-6} it is the master LP value, the dual-valid side, that is recorded as the node's bound in the tree, never the oracle value. The LP solver only ever sees this master, which has $n + n(n-1)/2 + S$ columns, $3n(n-1)/2$ McCormick rows and a few thousand active cuts.

Stabilization and engineering. Plain Kelley steps oscillate on this master. We generate cuts at the in-out point $z = \lambda x + (1-\lambda)c$ [4], where x is the master solution, c the center (best point so far) and $\lambda = 0.2$; if the cut at z does not cut off x a Kelley step at x follows. The center persists across nodes of the tree, projected onto the new box and the McCormick set, so each node starts from a point already near-optimal for its parent. Cut rows that have been basic and slack for more than five consecutive solves are deleted, which keeps the master at 1,000–4,000 rows. Each solve is a warm dual simplex from the previous basis with devex pricing (HiGHS 1.15.1 [8]); the solver state is reset before every solve, which costs nothing measurable and avoids a rare stall of warm re-solves.

Remark 6. *The decomposition computes the same bound as the monolithic LP* (12)*; it is a different way to compute it. Its advantage is that the growth of the cut pool never enters the LP the solver sees: new dual vertices only change the closed-form blocks.*

6 Branch-and-cut

Root. Seed the pool with the dual vertices at $y = \mathbf{0}$ and $y = \mathbf{1}$. Alternate: compute the bound (17) at the root box to tolerance; separate at its point by solving one recourse LP per factor ($S(1+2n)$ LPs, parallel over scenarios) and adding every violated vertex to the pool; repeat until nothing is violated. The bound is first computed to a loose tolerance 10^{-4} and tightened to 10^{-6} only once separation finds nothing. Rounding the root point gives the first incumbent.

Tree. Best-first on the parent bound; branching on the most fractional coordinate. At a node the bound runs from the persistent center with the node box and an *incumbent cutoff*: as soon as the master value exceeds the incumbent the node is pruned without finishing the bound, since the master value is a valid lower bound at every iteration.

Cuts are global. φ_s does not depend on the node box, so every scenario cut in the master and every dual vertex in the pool is valid at every node. Nothing is undone on backtracking.

Integer leaves. At a binary $\hat{y}$ the McCormick rows force $Y = \hat{y}\hat{y}^\top$, and the levels collapse: $L_j^+ = \hat{y}_j L_0$, $L_j^- = (1-\hat{y}_j)L_0$, so $\varphi_s(\hat{y}, \hat{y}\hat{y}^\top) = p_s(\hat{y})\, L_0$. The leaf is checked with S recourse LPs: if $Q_s(\hat{y}) > L_0$ for some s the new vertex is added and the node re-bounded from its center; otherwise $L_0 = Q_s(\hat{y})$ for all s, the node bound equals $F(\hat{y})$ by Proposition 1, and $\hat{y}$ updates the incumbent. This is the only place where exactness of the final answer is established.

Why fractional nodes do not separate. New dual vertices are separated only at the root (every factor, $S(1+2n)$ LPs per round) and at integer leaves (S LPs). On the 15 instances of Table 2 the root generates 9,200 to 46,100 vertices per instance and the whole tree adds 0 to 19 more; on nine instances the tree adds at most one. The root pool is in practice complete for the tree, because the lifted cuts of every factor at the converged root point already contain the vertices that are optimal in the neighbourhood the tree explores, and any vertex still missing at an integer leaf is caught there before the leaf can become an incumbent. Separating at fractional nodes would cost $S(1+2n)$ LPs per node for cuts the root has already found.

Heuristic and proof. Every fractional node point is rounded once per distinct rounded vector and evaluated exactly with S recourse LPs, whose optimal dual vertices go into the pool at no extra cost, so the pool holds an optimal vertex for every integer point the heuristic has visited; on our instances the optimum is found within the first 30 nodes and the rest of the

tree is proof. The run is proven when the heap is empty. Under a wall limit the smallest open bound is reported.

The McCormick solver used for comparison is the same tree code with the node bound replaced by the incremental MC-B LP, the same heuristic and the same tolerance. The two solvers differ only in the relaxation.

7 Computational results

All runs on one 8-core laptop (Apple silicon, 16 GB), C++ solver with HiGHS 1.15.1 for all LPs, eight threads for the recourse LPs, tolerance 10^{-6}, wall limits 7,200 s at 25–40 sites and 14,400 s at 50, idle machine. Three masters are run through the *same* branch-and-cut code, heuristic and tolerance: MC-B with the static bounds $[Q_s(\mathbf{1}),\ \text{pen}\sum_k d_{sk}]$ of Section 3; MC-B with *node-local* bounds, where at every node θ_s is bounded by $[Q_s(u),\ Q_s(\ell)]$ for the node box $\ell \le y \le u$ (monotonicity of Q_s in y), the four McCormick rows per (s, j) are rewritten with the node constants, and the recourse LPs this costs are charged to the run (a child changes one side of the box, so one of $Q_s(u)$, $Q_s(\ell)$ is inherited from the parent and $S = 20$ LPs are solved per node); and the decomposed linked master of Section 5. The McCormick masters separate Benders cuts once at every fractional node point and at integer points; the rounding heuristic and its dual vertices are shared by all three. Five seeds per size; the static master, which proves nothing above 35 sites, is run on seeds 1–3 only. The unlinked master of Section 4 (cuts (9)–(10) without rows (11)) is reported at the root only.

Table 1: Root relaxation gaps, $(F^* - v)/F^*$, of the three masters at the converged root LP, seeds 1–3; the unlinked master is the decomposed solver with the linking rows dropped (`--unlinked`).

n	seed	MC-B	unlinked	linked	n	seed	MC-B	unlinked	linked
25	1	8.29%	1.14%	0.37%	35	3	7.85%	0.60%	0.23%
25	2	7.19%	0.73%	0.14%	40	1	9.66%	0.36%	0.10%
25	3	7.86%	1.11%	0.31%	40	2	8.69%	0.40%	0.11%
30	1	6.75%	0.58%	0.29%	40	3	8.05%	0.87%	0.20%
30	2	7.76%	0.41%	0.07%	50	1	8.47%	0.68%	0.16%
30	3	6.71%	0.42%	0.22%	50	2	8.34%	0.26%	0.11%
35	1	6.13%	0.49%	0.14%	50	3	6.43%	0.46%	0.16%
35	2	8.09%	0.73%	0.43%					

Root gaps. Table 1 separates the two ingredients of the linked master. The lifting alone takes the root gap from 6–10% to 0.26–1.14%; the linking rows divide it by a further 1.7–6× (mean about 3×), to 0.07–0.43%. The ordering MC-B ≤ unlinked ≤ linked holds on every instance, as Proposition 2 requires. Node-local bounds do not change the root of MC-B: at the root $\ell = \mathbf{0}$, $u = \mathbf{1}$ and the two bound choices coincide.

Trees. The decomposed solver proves all 25 instances in tens to hundreds of nodes. Static MC-B needs thousands of nodes at 25 sites and tens of thousands from 30; it proves every 30-site instance (one in 75 minutes), two of three at 35, and none at 40 or 50, where after two and four hours its gaps are 1.5–3.4% and its best solution is not the optimum on three of the six instances. Node-local bounds are the fair comparison and they change the picture at small sizes: the McCormick tree shrinks by a factor of 4–8 against the static tree in the same code at 25 sites, and at 25–30 sites its time is within a factor of ten of the decomposed solver's on every seed, faster on two of ten. What node-local bounds do not change is the growth rate. The McCormick tree at 25, 30, 35, 40, 50 sites holds about 700, 1,500, 2,600, 11,000, 22,000 nodes on seeds 1–3 (unproven runs counted at the wall), a factor of about two per five sites, because

Table 2: Branch-and-cut: nodes and wall seconds to proven optimality, same C++ tree code for all three masters. "Unproven" gives the gap at the wall limit (7,200 s at 25–40 sites, 14,400 s at 50); the static master is not run on seeds 4–5. For the decomposed solver "root" is the time of the root cutting-plane loop, included in "time".

		MC-B, static		MC-B, node-local		decomposed		
n	seed	nodes	time (s)	nodes	time (s)	nodes	root (s)	time (s)
25	1	4,859	232	1,211	36	71	3	15
25	2	2,127	39	485	7	9	2	4
25	3	3,187	125	425	8	19	5	11
25	4			2,263	102	21	6	12
25	5			1,423	40	129	4	26
30	1	21,785	4,499	2,191	101	53	5	24
30	2	5,599	326	663	15	57	10	22
30	3	7,087	337	1,531	39	27	4	10
30	4			2,207	113	125	28	82
30	5			2,389	91	109	8	39
35	1	26,146	unproven (0.34%)	2,479	125	37	7	36
35	2	37,905	6,715	3,471	188	431	15	249
35	3	13,007	1,133	1,827	66	143	11	91
35	4			1,395	38	311	8	133
35	5			223	5	67	45	80
40	1	15,027	unproven (2.63%)	5,675	763	21	17	53
40	2	15,248	unproven (2.06%)	18,620	unproven (0.15%)	91	18	94
40	3	15,152	unproven (1.45%)	8,303	1,185	101	12	155
40	4			15,408	unproven (0.41%)	117	18	163
40	5			13,965	2,600	317	16	452
50	1	16,048	unproven (3.39%)	23,233	13,863	125	35	394
50	2	21,859	unproven (2.50%)	27,386	unproven (0.73%)	63	22	193
50	3	19,222	unproven (3.23%)	16,852	unproven (1.32%)	281	38	706
50	4			19,995	unproven (0.84%)	113	87	517
50	5			23,902	unproven (0.40%)	205	34	448

every node still starts from the 6–10% root gap and tightens only by branching; the decomposed tree stays in the tens to hundreds because its node gap is 0.1–0.5% before the first branch. At 40 sites the node-local master is 6–14× slower on the three seeds it proves and unproven after two hours on seeds 2 and 4 (gaps 0.15% and 0.41%); at 50 sites it proves seed 1 alone, in 13,863 s against 394 s, and leaves the other four unproven after four hours at 0.40–1.32%. It always holds the optimal solution; what it lacks is the proof. The node counts, not the wall times, are the cleanest statement of what the relaxation buys. Seeds 4 and 5 fall inside the ranges of seeds 1–3 for the decomposed solver (root gaps 0.08–0.55%, 12–517 s).

McCormick inside a full MIP solver. To rule out a weak implementation of the comparison, static MC-B was also handed to the HiGHS MIP solver: the MILP is built with the complete cut pool of the converged MC-B root LP of Table 1, solved to integer optimality, the integer point is separated with S recourse LPs, the new cuts are appended and the MILP re-solved (HiGHS has no lazy-constraint callback, so each round is a fresh MIP solve; 1,800 s per solve, 7,200 s wall). On the six 30- and 40-site instances of seeds 1–3 it proves all six in three to seven rounds: 180, 36 and 30 s at 30 sites and 234, 1,264 and 584 s at 40, against 24, 22, 10, 53, 94 and 155 s for the decomposed solver in Table 2. The MIP solver, with its own cutting planes, presolve and primal heuristics, closes the McCormick root gap two to twenty-five times faster than our node-local MC-B tree, and it still needs 1.6 to 13 times the wall time of the decomposed solver on every instance (2,328 s against 358 s summed), whose tree has none

of that machinery. Restarting the MIP every round discards its tree, which a lazy-constraint callback would keep: on the slowest instance the six MIP solves take 75 to 310 s each, every one a tree built from scratch at 40 sites. The experiment shows that a mature MIP code narrows the margin but does not remove it, because the gap it has to close by branching is the same 6.7–9.7% of Table 1 that the plain tree faces. Seeding the node-local McCormick master with the converged linked root pool instead of its own, as a referee suggested, changes nothing measurable: on the nine instances with 25 to 35 sites the McCormick root value is the same to the printed digits, because the converged McCormick root already satisfies every Benders cut at its own point and the linked pool contains no row the McCormick master can express, and the trees differ from the unseeded ones by at most 3% in nodes (2,157 against 2,191 on the first 30-site instance) and by a few seconds. The root gap, not the tree code, is what the McCormick master cannot overcome.

Lazy McCormick rows. The McCormick rows of the linked master (Section 4) can be added only when violated, like the lifted cuts. In the C++ code on the 50-site instance of seed 2 this takes the run from 277 s and 87 nodes to 226 s and 65 nodes with the same optimum and the same final row count (≈3,800); on the sparse 40-site instance below the row count falls from 986 to 899 at equal time. The rows are cheap and mostly tight, so the gain is modest.

Where the time goes. In the decomposed solver the master LP solve is 95–100% of the wall time on the 40- and 50-site instances of Table 2 (0.07–0.10 s per solve at $n = 40$, 0.15–0.20 s at $n = 50$, 12–24 solves per node), the separation with eight threads 0–5%, and the closed-form block evaluation is not measurable against them: the master LP is the cost.

Table 3: Sensitivity to the number of scenarios and to the endogeneity, $n = 30$, $m = 15$: decomposed solver (C++, two threads) and node-local MC-B (C++, S recourse LPs per node, wall 3,600 s); the $S = 20$ strong rows are the instances of Table 2. "Weak" endogeneity: uniform base probabilities, each opened site moving at most $0.9/(nS)$ of mass per scenario along a linear ramp across scenarios.

			decomposed			MC-B, node-local		
endogeneity	S	seed	root gap	nodes	time (s)	root gap	nodes	time (s)
strong	20	1	0.29%	63	36	6.75%	2,191	101
strong	20	2	0.07%	55	33	7.76%	663	15
strong	20	3	0.22%	25	13	6.71%	1,531	39
strong	50	1	0.26%	51	65	7.89%	1,239	164
strong	50	2	0.26%	83	56	8.77%	2,535	608
strong	50	3	0.24%	35	41	8.04%	1,599	239
strong	100	1	0.25%	159	196	8.41%	4,270	unproven (0.20%)
strong	100	2	0.18%	11	35	9.22%	3,189	2,235
strong	100	3	0.30%	37	65	7.49%	2,111	1,351
weak	20	1	0.18%	141	22	0.79%	425	5
weak	20	2	0.00%	0	0	0.00%	1	0
weak	20	3	0.19%	123	43	0.71%	1,317	37

Scenarios and endogeneity. Table 3 varies what the generator holds fixed elsewhere. The root gap is flat in S (0.18–0.30% at $S = 50$ and 100 against 0.07–0.29% at $S = 20$) and the time grows about linearly in S (means 27, 54, 99 s), as the per-scenario block structure of Section 5 predicts: the master gains S epigraph variables and the separation S recourse LPs, nothing quadratic. Weak endogeneity is easier for the bound, not harder: with little mass moving, the products $p_s(y)Q_s(y)$ are nearly linear and the root gap is at or below 0.2% and

even the McCormick gap falls to 0.8%; on seed 2 both roots are integral. The strong instances used throughout are the hard end of the generator. For the node-local McCormick master the scenario count is what hurts: its root gap stays at 7.5–9.2% and its time grows from 15–101 s at $S = 20$ to 1,351 s or more at $S = 100$, one instance unproven in an hour (gap 0.20%), against 35–196 s for the decomposed solver.

Table 4: Instances with $b_s = 0$ on the high-demand scenarios (generator option `--bpos0`; $m = 15$, $S = 20$, strong endogeneity otherwise unchanged), on which the linking rows of those scenarios are vacuous. Root gaps $(F^* - v)/F^*$ of the three masters at the converged root LP, and the trees of the decomposed solver and of node-local MC-B (wall 3,600 s; the McCormick tree is run at 25 and 30 sites). †Gap against the incumbent of an unproven run, an upper bound on the true gap.

		root gap			decomposed		MC-B, node-local	
n	seed	MC-B	unlinked	linked	nodes	time (s)	nodes	time (s)
25	1	23.95%	1.26%	1.20%	259	56	715	16
25	2	19.15%	1.17%	1.13%	83	21	2,235	105
25	3	23.87%	1.14%	1.05%	77	63	1,273	32
30	1	26.25%	1.79%	1.71%	237	595	1,697	79
30	2	32.26%	3.16%	3.05%	275	91	2,223	156
30	3	16.94%	0.63%	0.57%	273	122	9,738	unproven (0.74%)
35	1	22.83%	1.39%	1.32%	621	2,818		
35	2	27.44%	1.91%	1.83%	113	168		
35	3	25.19%†	1.77%†	1.70%†	537	unproven (0.23%)		
40	1	18.58%	0.83%	0.80%	771	2,379		
40	2	17.19%	0.34%	0.29%	135	199		
40	3	30.37%	1.44%	1.40%	415	2,834		

When the linking rows are vacuous. The linking rows tighten scenario s by exactly $\mathrm{bpos}_s(e_s^{0*} - L_0)$ (Remark 5), so a scenario with $\mathrm{bpos}_s = 0$ gains nothing from them. Table 4 takes this to the limit: the generator variant `--bpos0` gives the high-demand half of the scenarios no base mass at all, so their whole probability is moved in by opened sites and their linking rows are vacuous. Three things happen. The linked and unlinked roots nearly coincide, within 0.1 points on every instance, as the remark predicts. The factor cuts alone still hold the gap at 0.3–3.2%, against 17–32% for the McCormick master, whose global bounds are worst exactly when the probability of a scenario can swing from zero. And the gap is larger than on the standard instances (0.07–0.43%), so the decomposed tree is larger and slower: 77–771 nodes and 21 s to 47 min, one 35-site instance unproven in an hour at 0.23%. On a gap of this size the cheaper McCormick node pays: at 25 and 30 sites the node-local MC-B tree is faster on four of six instances (16–156 s against 21–595 s) and unproven after an hour on the sixth (9,738 nodes, 0.74%) where the decomposed solver needs 122 s; the McCormick tree was not run at 35 and 40 sites. These instances mark the edge of the method: its advantage is the root, and when the root gap is in the percent range rather than the tenth of a percent, the per-node master LP is expensive for what it closes at small n.

Customers. Table 5 varies the number of customers at 30 sites on fresh instances from the C++ generator (the $m = 15$ rows are therefore not those of Table 2). Each customer adds a row to every recourse LP and a coordinate to every dual vertex, and it is the vertex count that grows: the root pool holds 2–8 thousand vertices at $m = 15$ and 43–52 thousand at $m = 200$, the root loop takes 3–10 s against 203–285 s, and the separation itself, $S(1 + 2n)$ recourse LPs of size $n \times m$ per round, stays below 6% of the time at every m. The root gap grows with

Table 5: Sensitivity to the number of customers, $n = 30$, $S = 20$, strong endogeneity, fresh instances from the C++ generator, decomposed solver with eight threads, wall 3,600 s. "Pool" is the number of dual vertices in the pool at the end of the root loop, "added" the vertices the tree adds to it. †Gap against the incumbent of an unproven run, an upper bound on the true gap.

m	seed	root gap	root (s)	pool	added	nodes	time (s)
15	1	0.30%	10	8,041	56	111	61
15	2	0.00%	3	2,196	0	3	3
15	3	0.31%	7	3,791	329	233	92
50	1	0.50%	24	16,008	1,018	985	325
50	2	0.57%	58	19,635	456	131	142
50	3	0.78%	58	16,806	2,083	397	282
100	1	1.29%	167	31,585	23,328	3,485	1,950
100	2	0.44%	77	29,385	202	63	115
100	3	0.83%	58	20,716	4,293	781	382
200	1	1.24%†	203	42,640	53,934	3,858	unproven (0.18%)
200	2	1.09%	285	51,539	9,433	1,431	1,113
200	3	1.52%†	272	48,914	45,334	5,302	unproven (0.02%)

m as well, from at most 0.31% at $m = 15$ to 1.1–1.5% at $m = 200$: a recourse function with more pieces is less well approximated at the fractional root by cuts that are affine in the lifted variables. The consequence is in the tree. Up to 100 customers every instance is proven (3 to 3,485 nodes, 3 s to 33 min); at 200 one instance is proven in 19 min and two are unproven after an hour at 0.18% and 0.02%, with 45–54 thousand vertices added to the pool in the tree. The time goes where it always does: on the two unproven instances 3,446 and 3,397 of the 3,600 s are master LP solves, 20–30 per node, since the node cutting-plane loop needs more iterations on a larger gap. The number of customers reaches the solver through the gap and the pool, not through the size of the recourse LPs.

8 Scaling and parallelism

The dense lifting is quadratic in n: every cut row is about 60% dense over the $n(n-1)/2$ pair columns, so master nonzeros grow with n^2 and the simplex iteration cost with them (Table 6). That is the one cost a larger machine does not remove, and it sets what the dense master can do on a laptop.

Table 6: Dense master size at 4,000 rows; measured at $n = 50$, extrapolated on the 30–50-site trend. At $n = 100$ the measured root takes 15,200 rows and 11–21 minutes.

sites	pairs	nnz/cut row	master nnz	memory/rank	s/solve
50 (measured)	1,225	720	2.9 M	40 MB	0.15–0.20
100	4,950	2,900	12 M	150 MB	1–3
200	19,900	11,700	47 M	600 MB	10–25
500	124,750	73,000	290 M	3.5 GB	60–250

Node-parallel tree. The tree parallelizes well: nodes are independent given the shared pool, and the few cuts a node generates (only at integer leaves and rounded points) are valid everywhere. The node-parallel implementation is an MPI program in which rank 0 computes the root once with all eight separation threads, then broadcasts the dual-vertex pool, the active master cut rows and the stabilization center (two megabytes at 50 sites) so that every worker

starts from the converged root state; rank 0 then keeps the best-first heap, the incumbent and the global pool, and each other rank owns a master LP and a replica of the pool, receives a node box together with any cuts it has not yet seen, bounds it with the machinery of Section 5, and returns the branching variable, any incumbent and any new cuts. Table 7 gives the measurements with one coordinator and seven workers on the otherwise idle 8-core laptop, against the single-rank code using all eight cores for separation. The root costs the same in both (the idle ranks must be told to yield the processor while they wait for the broadcast, otherwise their polling slows the root by half); the tree speedup on seven workers is 2.1–4.0×, lowest on the trees of 25–91 nodes, where best-first search rarely has seven nodes of comparable bound open at once, and 3.2–4.0× on the 105–269-node trees. The net effect is 1.6–3.3× end to end, growing with the tree, which is the regime where parallelism is needed.

Table 7: MPI node-parallel tree, one coordinator and seven workers on an idle 8-core machine, root computed once on the coordinator, against the single-rank code with eight separation threads. Same optima, all proven; "tree" is the wall time after the root.

		single rank			1+7 ranks			speedup	
n	seed	nodes	root (s)	tree (s)	nodes	root (s)	tree (s)	tree	total
40	1	21	17	36	25	16	17	2.1×	1.6×
40	2	91	18	76	91	20	36	2.1×	1.7×
40	3	101	12	143	105	12	37	3.9×	3.2×
50	1	125	35	359	113	29	92	3.9×	3.3×
50	2	63	22	171	63	24	54	3.2×	2.5×
50	3	281	38	668	269	43	169	4.0×	3.3×

Dense 100 sites. The C++ root on the three dense 100-site instances closes in 1,185, 680 and 1,277 s at 15,200–15,300 master rows, with root gaps to the best known solutions of 0.08%, 0.04% and 0.11%: the gap does not widen with n at fixed S and m. The tree is another matter. On seed 1 with one coordinator and seven workers the search reached 100 nodes after 79 minutes with 89 nodes still open and a gap of 0.058% (bounds 71.3123 and 71.3536), and was stopped. At a node gap that small many boxes tie, the best-first heap widens, and each node still costs a 15,000-row master solve; closing a dense 100-site instance is a multi-hour job on eight cores and we report it as the practical limit of the dense master on this hardware, not as a solved row.

Sparse influence at 100–500 sites. At 500 sites with every site influencing demand, one facility shifts 0.2% of the probability mass, which is not a model anyone would build; the realistic structure is *sparse influence*, a few hubs that change the demand regime. The sparse master of Section 4 keeps only (influencing)×(any) pair columns. Table 8 uses the same generator with an influence density ρ chosen so that about ten sites influence demand at each size ($\rho = 0.1$, 0.05, 0.02); the sparse master then has $O(|\mathcal{I}|n)$ pair columns instead of $O(n^2)$ and its S blocks cost $O(S|\mathcal{I}|n)$ to evaluate instead of $O(Sn^2)$, and 500 sites is a 26–222 s job on the laptop with a root gap below 0.005%. The dense and sparse masters give the same optimum where both were run (the 100-site instance of seed 1: dense root 28 s and 14,983 rows at a 0.07% gap, sparse root 1 s and 2,164 rows at 0.08%). The last row raises the number of customers from 15 to 100: the root gap is still below 0.005% and the root takes 108 s, but each node then costs about 1.5 s of recourse LPs and the tree is wide (2,865 nodes in two hours, gap 0.003% at the wall), so the recourse, not the master, becomes the cost. MC-B has only nS bilinear terms and stays small at any size, so it remains the comparison at every scale; its root gap is unchanged by sparsity.

Table 8: Sparse-influence instances, $m = 15$, $S = 20$, about ten influencing sites per instance; decomposed C++ solver with the sparse master, eight threads, wall 7,200 s.

n	ρ	seed	optimum	root gap (s)	nodes	time (s)	master rows
100	0.10	1	47.829147	0.08% (1)	255	8	2,164
100	0.10	2	46.275180	0.04% (1)	121	4	2,184
100	0.10	3	50.156085	0.04% (1)	43	6	3,610
200	0.05	1	46.184683	0.01% (3)	35	8	6,112
200	0.05	2	46.601454	0.01% (3)	2,155	114	4,901
200	0.05	3	47.438137	0.01% (3)	667	59	4,448
500	0.02	1	47.766355	<0.005% (12)	57	26	12,207
500	0.02	2	46.519291	<0.005% (31)	385	222	20,922
500	0.02	3	45.413226	<0.005% (17)	167	82	19,723
500, m=100	0.02	1	262.941205	<0.005% (108)	2,865	unproven (0.003%)	19,788

9 Related work

Dupačová's classification separates exogenous from endogenous uncertainty and, within the latter, the case of a fixed scenario set with decision-dependent weights; she notes that convexity in the decision is lost and points to enumeration and branch-and-bound [5]. Hellemo, Barton and Tomasgard formulate exactly the scenario-weight model (4) with linear-scaling and convex-combination forms of $p_s(y)$ and solve the bilinear programs with a global solver [7]. Ahmed casts discrete first-stage problems with decision-dependent uncertainty as 0–1 hyperbolic programs [2]. Basciftci, Ahmed and Shen linearize binary-times-bounded-continuous products with the McCormick set and note its exactness on binary points [3]; that is MC-B. Pantuso's facility-location model with combinatorially many decision-dependent distributions enumerates distribution patterns and linearizes an indicator times the second-stage cost [6], and Pantuso and Hewitt's L-shaped method for decision-dependent uncertainty uses McCormick envelopes for a distribution-independent LP bound [9]. In terms of scale, the exact methods above report at most 10 candidate sites in a monolithic McCormick MILP [3] and, for the pattern-enumeration model, 20–25 candidate sites with 10–15 zones that select the distribution (2^{10}–2^{15} patterns) in minutes to two hours, with 17 zones solved partly and 20 not at all in ten hours [6]. The models differ from (4), so Section 7 is a comparison of scale and structure, not of methods on common instances: there every one of 25–50 sites influences the probabilities, and the sparse 500-site instances have about ten influencing sites. The lifting of Section 4 is the mixed-binary reformulation–linearization of Adams and Sherali [1] on the dual side of the recourse; the in-out stabilization is that of Ben-Ameur and Neto [4].

10 Conclusion

A decision-dependent probability does not have to be relaxed through global bounds on the recourse. Written as a posiform, each of its terms multiplies a recourse LP by a binary variable, which is again a recourse LP at a scaled right-hand side, and the shared dual polyhedron turns every dual vertex into a cut that is linear in the lifted variables. The resulting LP is nearly tight at the root, decomposes by scenario with closed-form blocks, and its cuts survive the whole branch-and-bound tree. On 25 instances with 25 to 50 sites it proves optimality in minutes where the McCormick master in the same tree code does not finish in hours, and 1.6–13× faster than the McCormick master inside a full MIP solver. The fair McCormick baseline, with recourse bounds recomputed at every node, is competitive up to 35 sites and then falls behind by a factor that doubles every five sites, because its root gap is unchanged. The limit of the method is the same quantity seen from the other side: on instances where the linking rows are vacuous ($b_s = 0$) or with 200 customers, the root gap is in the percent range, the tree grows to thousands

of nodes, and at 25–30 sites the cheaper McCormick node is competitive again (Tables 4 and 5).

The affine probability model is a modelling choice with limits: the influence of the sites is additive and must remain a probability at every configuration, so it cannot express saturation or thresholds, which the conditional distributions of [6] can. The construction does not depend on affinity, only on p_s being a polynomial in y: a multilinear p_s, which is every function of binary y, has a posiform in products of literals, each product times $Q_s(y)$ is again the recourse LP at a right-hand side affine in the corresponding higher-order lifted variable, and the same dual vertices give the cuts, at the price of level-k RLT lifting. We have not pursued this. The quadratic lifting bounds dense instances near 100 sites on a laptop; with sparse influence, the realistic structure at that scale, the same solver proves 500-site instances in minutes.

Declaration on the use of AI assistance

This work was carried out with Claude (Anthropic), used as an AI assistant throughout the project: for the cross-field literature survey from which the lifted-cut construction was selected, for writing the C++ solver and the experiment scripts, for running and recording the computational study, and for drafting and revising this text, including the responses to review. The research question, the choice of method, the standards for the experiments and the final decisions on every claim are the author's. The author reviewed and edited all generated content, verified the proofs and the reported numbers against the solver's output, and takes full responsibility for the content of the publication. The source code, the instances and the result files that support every number in the paper are provided with it.

References

[1] W. P. Adams and H. D. Sherali. Linearization strategies for a class of zero-one mixed integer programming problems. *Operations Research* 38(2):217–226, 1990.

[2] S. Ahmed. *Strategic planning under uncertainty: stochastic integer programming approaches.* PhD thesis, University of Illinois at Urbana-Champaign, 2000.

[3] B. Basciftci, S. Ahmed and S. Shen. Distributionally robust facility location problem under decision-dependent stochastic demand. *European Journal of Operational Research* 292(2):548–561, 2021.

[4] W. Ben-Ameur and J. Neto. Acceleration of cutting-plane and column generation algorithms: applications to network design. *Networks* 49(1):3–17, 2007.

[5] J. Dupačová. Optimization under exogenous and endogenous uncertainty. In *Proceedings of MME 2006*. `https://www2.karlin.mff.cuni.cz/~kopa/papers/MME06sty.pdf`

[6] G. Pantuso. Solution of stochastic facility location problems with combinatorially many decision-dependent distributions. *Journal of Combinatorial Optimization*, 2025. doi:10.1007/s10878-025-01371-7.

[7] L. Hellemo, P. I. Barton and A. Tomasgard. Decision-dependent probabilities in stochastic programs with recourse. *Computational Management Science* 15:369–395, 2018.

[8] Q. Huangfu and J. A. J. Hall. Parallelizing the dual revised simplex method. *Mathematical Programming Computation* 10:119–142, 2018.

[9] G. Pantuso and M. Hewitt. The L-shaped method for stochastic programs with decision-dependent uncertainty. *Mathematical Programming*, 2025. doi:10.1007/s10107-025-02246-9.