OPTIMIZATION FOR ONE, FEASIBILITY FOR MOST: DECOUPLING OPTIMIZATION FROM FEASIBILITY IN CHANCE-CONSTRAINED PROGRAMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Chance-constrained stochastic programs (CCSPs) require solutions that satisfy uncertain constraints with high probability. Standard approaches either embed scenarios directly into optimization by considering sample average approximation (SAA)-based reformulations, growing prohibitively expensive and often failing to find feasible solutions; or employ conservative reformulations that guarantee feasibility but sacrifice objective quality by design. We propose ACCESS (Adaptive Chance-Constrained Exploration via Structured Sampling), a structured sampling framework that decouples optimization from feasibility evaluation. First, high-quality candidate solutions are generated by solving many inexpensive single-scenario subproblems, avoiding sampling directly in the constrained high-dimensional solution space. Second, feasibility is assessed separately via large-scale batched simulation over a massive number of scenarios, which is naturally amenable to GPU parallelism. That reduces the search from a high-dimensional solution space to a two-dimensional parameter space controlling *conservatism* and *scenario diversity*; while concentrating on candidates near the feasibility boundary, where high-quality feasible solutions lie. We also introduce a variant, ACCESS(δ), which provides certified feasibility guarantees with probability $1-\delta$. Experiments across three benchmark classes and five scenario distributions show that ACCESS matches or exceeds the best objective found by any method in nearly every setting, while ACCESS(δ) is empirically feasible on every instance. The conservative reformulations remain several percent worse in objective, and SAA is frequently infeasible while taking an order of magnitude longer on the harder benchmarks.

1 INTRODUCTION

Real-world decision-making under uncertainty routinely demands solutions that are reliable with high probability. For instance, a production planner must ensure supply meets demand in at least 90% of scenarios; an energy grid operator must maintain feasibility across 95% of demand realizations; a logistics provider must meet delivery commitments in all but rare cases (Shapiro et al., 2009; Küçükyavuz & Jiang, 2022). These settings are naturally modeled as *chance-constrained stochastic programs* (CCSPs), which require finding a solution $\mathbf{x}$ from a set $\mathcal{X} \subseteq \mathbb{R}^n$ that satisfies a constraint with uncertain parameters with confidence level $\alpha \in (0, 1)$; which is chosen as a large value by risk-averse decision makers, e.g., $\alpha \geq 0.90$. CCSPs are computationally intractable as its feasible regions are nonconvex and feasibility is expensive to evaluate except for special cases. They become especially challenging when the decisions become combinatorial, i.e., $\mathbf{x} \in \mathcal{X} \subseteq \{0, 1\}^n$ (binary) or $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{Z}^n$ (integer). Hence, we evaluate feasibility empirically and refer to a solution as α-*feasible* if it satisfies all uncertain constraints in at least an α fraction of the evaluation scenarios.

The dominant approach, *sample average approximation* (SAA), replaces the probability in the chance constraint with an empirical estimator based on an independently and identically distributed (i.i.d.) set of scenarios and solves a monolithic large-scale optimization problem consisting of a mixed-integer linear program (MILP) as its approximate representation. SAA enjoys strong theoretical guarantees (Luedtke & Ahmed, 2008; Pagnoncelli et al., 2009), but its computational cost grows rapidly with the size of the sample set and the problem size. Reliably estimating a 99th-percentile constraint requires thousands of scenarios for discerning between feasible and infeasible solutions around the

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

optimal value, making the corresponding MILP intractable for combinatorial problems of practical interest. An alternative class of methods, safe approximations, replaces the chance constraint with a tractable convex surrogate that guarantees feasibility by construction (Rockafellar & Uryasev, 2000; Nemirovski & Shapiro, 2006; Chen et al., 2007). However, this conservatism comes at a cost as solutions are often suboptimal. Both SAA and safe approximations are *single-shot* methods, i.e., they solve a single optimization problem and thus produce a single solution.

We take a fundamentally different approach. Rather than solving once, we *sample* many candidate solutions at varying conservatism levels and select the best feasible one. Our framework, ACCESS (Adaptive Chance-Constrained Exploration via Structured Sampling), decouples optimization from feasibility evaluation. Rather than embedding scenarios into a monolithic optimization problem, we solve inexpensive single-scenario subproblems parameterized by two scalars: (1) a conservatism level γ that controls how tightly constraints are enforced, and (2) a scenario magnitude β that interpolates between nominal (deterministic) and diverse sampled scenarios. Each (γ, β) pair yields a candidate solution and feasibility is then assessed separately via batched evaluation over a large validation set. Bayesian Optimization (BO) navigates this 2D parameter space to concentrate candidates near the feasibility boundary, where high-quality feasible solutions lie. This approach exploits parallelism at two levels: many small subproblems can be solved independently across CPU cores, and feasibility evaluation over millions of scenarios parallelizes efficiently on GPUs for rapid best-of-N selection.

Figure 1 illustrates this on instances of the chance-constrained knapsack problem. In (a), SAA reaches high objective values but often violates the chance constraint whereas conservative methods are feasible but lower in objective, whereas ACCESS bridges the gap between them. In (b), the candidates produced by random (γ, β) pairs trace the trade-off between solution feasibility and quality, with the search in ACCESS concentrated on candidates where that trade-off meets α-feasibility.

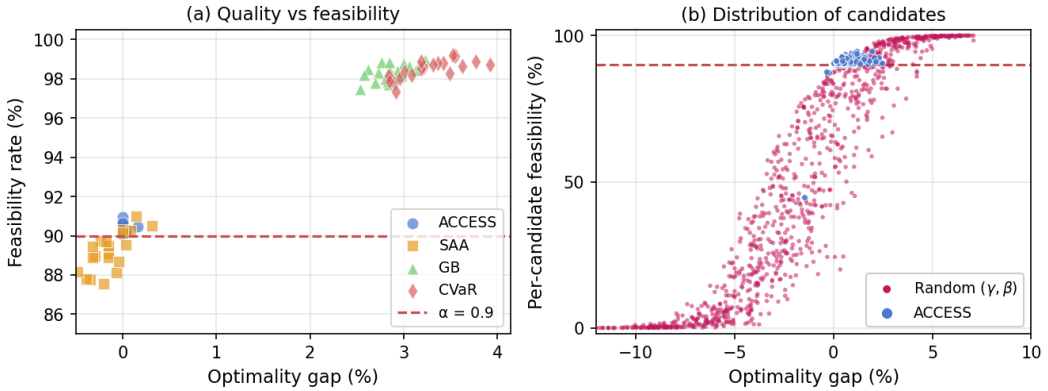

Figure 1: **(a)** Optimality gap versus feasibility rate on the largest knapsack instances at α=0.90, one point per instance. Gap is relative to the best α-feasible objective found by any method, so a solution that beats it while violating the chance constraint appears at a negative gap. GB and CVaR denote the Gaussian–Bonferroni and CVaR safe approximations, respectively; all baselines are introduced in Section 4 and Appendix B **(b)** Feasibility and optimality gap of 1,000 candidates from random (γ, β) pairs and of the 1,000 ACCESS candidates, on a single knapsack instance.

Contributions.

1. **A two-parameter subproblem family, searched with BO.** We define a family of single-scenario MILP subproblems parameterized by a tightening parameter γ and a diversification parameter β, whose endpoints are the nominal scenario used in robust approximations (β=0) and each realized scenario (β=1). For each scenario, constrained BO searches the space (γ, β), leaving the combinatorial structure to the solver and concentrating candidates near the feasibility boundary.
2. **Distribution-free feasibility guarantees.** In classic scenario-based methods, e.g., SAA/CVaR, the scenarios that back the feasibility guarantee sit inside the optimization model, so their number is limited by what the solver can handle. ACCESS moves the guarantee to the selection step: candidates are generated without the scenarios on which they are selected, and checking a candidate is cheap and amenable to GPU parallelism, providing strong guarantees that can be computed

efficiently. In addition, this yields a certified variant, ACCESS(δ), that returns an α-feasible solution with probability at least $1 - \delta$ for any distribution that can be sampled.

3. **Comprehensive evaluation.** Across knapsack, matching and shortest-path benchmarks under five distributions, ACCESS and ACCESS(δ) return α-feasible solutions at a worst-case mean gap of 0.04% and 0.64% to the best objective found by any method, while SAA is frequently infeasible and the safe approximations are overly conservative. Moreover, solutions are found in a fraction of the time SAA requires, demonstrating the computational efficiency of the proposed approach.

2 PRELIMINARIES

2.1 CHANCE-CONSTRAINED STOCHASTIC PROGRAMS

A *chance-constrained stochastic program* (CCSP) takes the form

$$\max_{\mathbf{x}\in\mathcal{X}} \quad f(\mathbf{x}) \tag{1a}$$

$$\text{s.t.} \quad \Pr_{\boldsymbol{\xi}}(\mathbf{x} \in \mathcal{P}(\boldsymbol{\xi})) \geq \alpha, \tag{1b}$$

where $\mathcal{X} \subseteq \mathbb{Z}^p \times \mathbb{R}^{n-p}$ is a deterministic feasible set on integer and continuous decision variables, $f : \mathcal{X} \to \mathbb{R}$ is the objective function, and the chance constraint represents the set of feasible solutions $\mathcal{P}(\boldsymbol{\xi})$ under a random vector $\boldsymbol{\xi}$ with probability distribution $\mathbb{P}$ and confidence level $\alpha \in (0, 1)$. In this work, we mainly focus on linear chance constraints by considering a polyhedral representation $\mathcal{P}(\boldsymbol{\xi}) := \{\mathbf{x} : \mathbf{A}(\boldsymbol{\xi})\mathbf{x} \leq \mathbf{b}(\boldsymbol{\xi})\}$, where $\mathbf{A}(\boldsymbol{\xi}) \in \mathbb{R}^{d\times n}$ is a coefficient matrix, in which the i-th row is $\mathbf{a}_i(\boldsymbol{\xi}) \in \mathbb{R}^n$, and $\mathbf{b}(\boldsymbol{\xi}) \in \mathbb{R}^d$ is the vector of right-hand sides. For the nominal data, we use $\bar{\mathbf{A}} = \mathbb{E}_{\boldsymbol{\xi}}[\mathbf{A}(\boldsymbol{\xi})]$, in which the i-th row is $\bar{\mathbf{a}}_i$, and $\bar{\mathbf{b}} = \mathbb{E}_{\boldsymbol{\xi}}[\mathbf{b}(\boldsymbol{\xi})]$. This setting covers a broad class of optimization problems, such as knapsack, matching, and routing; and allows us to consider both individual ($d = 1$) and joint chance constraints ($d > 1$), with all d constraints holding *simultaneously*.

Solution approaches. In the widely used Sample Average Approximation (SAA), the probability over $\boldsymbol{\xi}$ is replaced with an empirical estimate over a set of scenarios $\mathcal{S} := \{\boldsymbol{\xi}^{(i)}\}_{i=1}^{|\mathcal{S}|} \overset{\text{i.i.d.}}{\sim} \mathbb{P}$, yielding

$$\max_{\mathbf{x}\in\mathcal{X}} \left\{ f(\mathbf{x}) : \frac{1}{|\mathcal{S}|} \sum_{i=1}^{|\mathcal{S}|} \mathbb{I}\left\{ \mathbf{A}(\boldsymbol{\xi}^{(i)})\mathbf{x} \leq \mathbf{b}(\boldsymbol{\xi}^{(i)}) \right\} \geq \alpha \right\}. \tag{2}$$

The indicator $\mathbb{I}$ equals one when $\mathbf{x}$ satisfies all constraints in a given scenario and zero otherwise. Hence, the SAA formulation requires simultaneous satisfaction of all constraints in at least $\lceil \alpha|\mathcal{S}| \rceil$ sampled scenarios. For the linear models considered in this paper, the polyhedral structure of $\mathbf{A}(\boldsymbol{\xi}^{(i)})\mathbf{x} \leq \mathbf{b}(\boldsymbol{\xi}^{(i)})$ allows equation 2 to be reformulated as an equivalent MILP using binary variables and valid big-M constraints, as detailed in Appendix B.1.

Safe approximations replace the chance constraint equation 1b with deterministic sufficient conditions, often expressed as convex constraints, yielding

$$\max_{\mathbf{x}\in\mathcal{X}} \left\{ f(\mathbf{x}) : \mathbf{x} \in \mathcal{X}_{\text{safe}} \right\}, \tag{3}$$

where $\mathcal{X}_{\text{safe}} \subseteq \{\mathbf{x} \in \mathcal{X} : \Pr_{\boldsymbol{\xi}}(\mathbf{x} \in \mathcal{P}(\boldsymbol{\xi})) \geq \alpha\}$. Thus, every feasible solution of the safe approximation is feasible for the original problem, but some feasible solutions of the original problem may be excluded ($x \in \mathcal{X} : \Pr_{\boldsymbol{\xi}}(\mathbf{x} \in \mathcal{P}(\boldsymbol{\xi})) \geq \alpha \not\Rightarrow x \in \mathcal{X}_{\text{safe}}$); thereby affecting the quality of the solution obtained. Examples of safe approximations include CVaR and Bernstein approximations, as well as approaches that combine individual safe approximations through Bonferroni inequalities. We use two representative safe approximations, Gaussian–Bonferroni (GB) and CVaR, as benchmarks for our method. Details are provided in Section 3.1 and Appendix B.2.

Evaluating feasibility. Computing the exact probability in equation 1b is generally intractable. Hence, we use an empirical feasibility rate defined over a set of scenarios $\mathcal{T} := \{\boldsymbol{\xi}^{(i)}\}_{i=1}^{|\mathcal{T}|} \overset{\text{i.i.d.}}{\sim} \mathbb{P}$ as

$$p_{\mathcal{T}}(\mathbf{x}) = \frac{1}{|\mathcal{T}|} \sum_{i=1}^{|\mathcal{T}|} \mathbb{I}\left\{ \mathbf{A}(\boldsymbol{\xi}^{(i)})\mathbf{x} \leq \mathbf{b}(\boldsymbol{\xi}^{(i)}) \right\}, \tag{4}$$

and consider $\mathbf{x}$ to be α-*feasible* if $p_{\mathcal{T}}(\mathbf{x}) \geq \alpha$. This is standard practice in stochastic optimization.

We note that this is the same approach used in SAA, but sets $\mathcal{S}$ and $\mathcal{T}$ are used for different purposes and at different times: the first to find a solution and the second to evaluate if the solution is indeed feasible. Drawing a parallel with machine learning tasks, $\mathcal{S}$ and $\mathcal{T}$ are akin to training and test sets.

Throughout the paper, multiple scenario sets such as $\mathcal{S}$ and $\mathcal{T}$ appear. Each is an independent i.i.d. sample from $\mathbb{P}$, but to simplify notation we uniformly refer to their elements as $\xi^{(1)}, \xi^{(2)}$, and so on.

2.2 Related Work

Optimization under uncertainty. Chance-constrained programs are challenging to solve due to expensive feasibility verification and nonconvex, possibly disconnected feasible regions (Nemirovski & Shapiro, 2006; Ahmed & Papageorgiou, 2013; Küçükyavuz & Jiang, 2022). Two main approaches exist: convex conservative approximations (Nemirovski & Shapiro, 2006; Chen et al., 2007; Rockafellar & Uryasev, 2000) and SAA (Luedtke & Ahmed, 2008). Both are single-shot methods with fixed conservatism levels. We build on these quantile-type approximations but treat their conservatism constant as searchable, enabling empirical navigation of the feasibility-optimality tradeoff. Our parameterized search also relates to robust optimization (Ben-Tal et al., 2009; Bertsimas & Sim, 2004), where a scalar controls uncertainty-set size. Closest to us, Li & Li (2015) and Yuan et al. (2017) also tighten the nominal formulation and search for the margin rather than fixing it with a bound, differing in how that search is carried out. Both return a single solution, whereas we sample the scenario that defines each subproblem and certify the one we select.

Learning to optimize. Learning-based methods for combinatorial optimization have largely focused on improving optimization algorithms, for example through learned branching policies (Khalil et al., 2016; Gasse et al., 2019) or neural solution construction methods (Kool et al., 2019). More closely related to our work are recent sampling-based approaches, e.g., (Sun & Yang, 2023; Sun et al., 2023; Ichikawa & Arai, 2025), which generate diverse candidate solutions through stochastic search dynamics directly in the combinatorial solution space. Constraints enter these methods as penalties in the sampling dynamics, so α-feasibility would require a differentiable surrogate for the empirical feasibility rate equation 4. They also target problems where the MILP solver itself is the bottleneck, whereas in a CCSP the bottleneck arises from coupling the scenarios in one model, as in SAA, and solving many deterministic subproblems independently is comparatively cheap.

Constraint learning. A related direction embeds learned predictive models directly into MILP formulations (Maragno et al., 2025). Building on this paradigm, Alcántara & Ruiz (2023) propose chance-constraint learning via conditional quantile models, assuming access to data from the underlying stochastic process. Learn-and-embed methods for stochastic and bilevel optimization (Patel et al., 2022; Dumouchelle et al., 2024) similarly train value-function surrogates and embed them in optimization models, with training data generated by sampling decisions randomly. Recent work addresses the resulting data-collection issue by alternating surrogate optimization and retraining on samples near incumbent solutions (Kronqvist et al., 2023). In contrast, our framework does not learn an embedded surrogate or use iterative active data collection; instead, it directly generates boundary-focused candidate solutions through structured problem-space sampling. This distinction matters in combinatorial CCSPs since the training data must be generated, and random solutions rarely fall near the feasibility boundary (Appendix H.3), so a learned constraint would first need a boundary-focused sampler, which is the problem ACCESS addresses.

3 Methodology

ACCESS decouples optimization from feasibility evaluation: instead of solving one analytically tightened problem, it *searches* empirically over subproblems of varying conservatism (γ) and diversity (β), as outlined in Figure 2. In what follows, we describe the subproblem family (Section 3.1), searching over it (Section 3.2), batched evaluation (Section 3.3), and a feasibility guarantee (Section 3.4).

3.1 Parameterized Subproblems

ACCESS solves parameterized subproblems resembling two standard surrogates of the CCSP, the single-scenario problem and the safe approximation, to balance their opposite drawbacks.

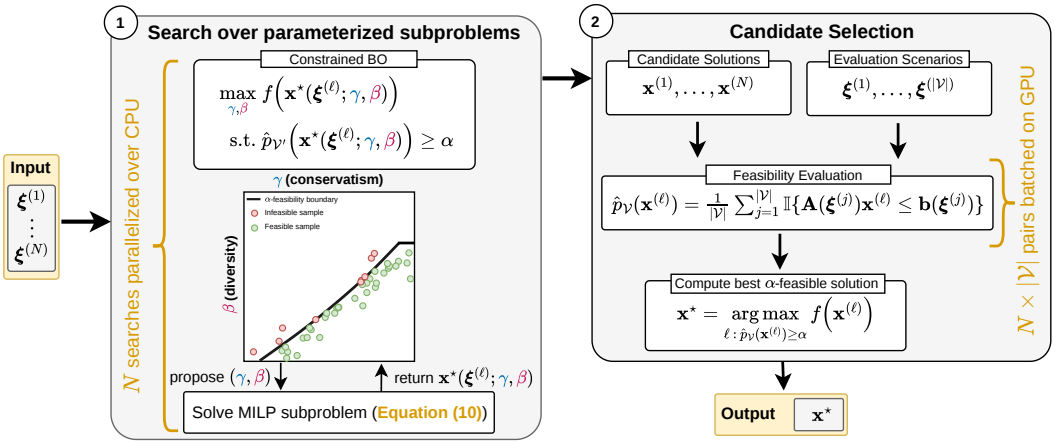

Figure 2: The ACCESS pipeline. (1) For each of the N sampled scenarios, constrained BO searches over (γ, β), in parallel on CPU. (2) The resulting candidates $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}$ are scored against a set of validation scenarios $\mathcal{V}$ in one batched GPU pass, and the best α-feasible solution is returned.

Single-scenario subproblems (and SAA). A natural heuristic is to sample a scenario $\boldsymbol{\xi} \sim \mathbb{P}$ and solve the deterministic problem induced by the realization of this scenario, i.e.,

$$\mathbf{x}^\star(\boldsymbol{\xi}) \in \underset{\mathbf{x}\in\mathcal{X}}{\arg\max} \{f(\mathbf{x}) \ : \ \mathbf{A}(\boldsymbol{\xi})\mathbf{x} \leq \mathbf{b}(\boldsymbol{\xi})\}. \tag{5}$$

Each deterministic solve is cheap and each draw $\boldsymbol{\xi}$ gives a different solution $\mathbf{x}^*(\boldsymbol{\xi})$, but this solution is tailored to its scenario: some constraints might be binding ($\mathbf{a}_i(\xi)^T\mathbf{x}^*(\boldsymbol{\xi}) = b_i$ for some i), especially if the decision variables are continuous, and those likely violate the chance constraint under $\mathbb{P}$. This is addressed in SAA by ensuring that the solution is feasible across a ratio α of a set of scenarios, but theoretical guarantees require a number of scenarios that is a computationally prohibitive.

Safe approximations. Safe approximations instead solve a conservative formulation over the nominal problem. Under multiplicative uncertainty, let us consider the component-wise setting in which $a_{i,j}(\boldsymbol{\xi}) = \bar{a}_{i,j}(1 + \sigma_i\xi_j)$, where $\bar{a}_{i,j} = E_{\boldsymbol{\xi}}[a_{i,j}(\boldsymbol{\xi})]$ and σ_i is the relative standard deviation shared by all coefficients of row i. Using the element-wise (Hadamard) product, we denote the left-hand side of each constraint i as $\mathbf{a}_i(\boldsymbol{\xi}) = \bar{\mathbf{a}}_i \odot (\mathbf{1} + \sigma_i\boldsymbol{\xi})$, where $\mathbf{1}$ denotes the all-ones vector and $\boldsymbol{\xi} \in \mathbb{R}^n$ carries a noise component per decision variable. For a fixed $\mathbf{x} = \tilde{\mathbf{x}}$, the left-hand side of constraint i is a random variable with mean $\bar{\mathbf{a}}_i^\top\tilde{\mathbf{x}}$ and standard deviation $\sigma_i\|\bar{\mathbf{a}}_i \odot \tilde{\mathbf{x}}\|_2$. In the safe approximation, we solve for $\mathbf{x}$ subject to

$$\bar{\mathbf{a}}_i^\top\mathbf{x} + \kappa\,\sigma_i\|\bar{\mathbf{a}}_i \odot \mathbf{x}\|_2 \leq \bar{b}_i, \qquad \forall i = 1,\ldots,d, \tag{6}$$

where κ is selected to ensure feasibility under the scenario distribution $\mathbb{P}$ (see Appendix B.2). This approach is elegant for relying on a single parameter κ thanks to σ_i allowing us to scale the margin across constraints, but it is bound to yield suboptimal solutions because it is excessively conservative.

Striking a balance. ACCESS is derived from the approaches above in three steps. First, we circumvent the caveat of the safe approximation by working with an empirical parameter γ replacing κ, meaning that we will search for the least conservative γ whose solution remain α-feasible on a large set of sampled scenarios. Because the norm in equation 6 makes the subproblem a mixed-integer second-order cone program, we mitigate the cost of solving this subproblem multiple times by using the $L1$-norm instead. For simplicity, we assume $\mathbf{A}$ and $\mathbf{x}$ are non-negative, as they are in our benchmarks and in packing and covering problems generally. Under this assumption, $\|\bar{\mathbf{a}}_i \odot \mathbf{x}\|_2 \leq \|\bar{\mathbf{a}}_i \odot \mathbf{x}\|_1 = \bar{\mathbf{a}}_i^\top\mathbf{x}$, so replacing the norm by the linear term only tightens the constraint and leaves a linear capacity, $(1 + \kappa\sigma_i)\,\bar{\mathbf{a}}_i^\top\mathbf{x} \leq \bar{b}_i$, which is still a valid safe approximation in which

$$(1 + \gamma\sigma_i)\,\bar{\mathbf{a}}_i^\top\mathbf{x} \leq \bar{b}_i \qquad \text{or, equivalently,} \qquad \bar{\mathbf{a}}_i^\top\mathbf{x} \leq \frac{\bar{b}_i}{1 + \gamma\sigma_i} \qquad \forall i = 1,\ldots,d. \tag{7}$$

Second, we produce multiple solutions instead of a single one by solving the subproblem on a set of scenarios $\{\xi^{(\ell)}\}_{\ell=1}^N$ instead of only the nominal case. For each scenario ξ, the subproblem becomes

$$\mathbf{x}^\star(\boldsymbol{\xi};\gamma) \in \underset{\mathbf{x}\in\mathcal{X}}{\arg\max} \left\{ f(\mathbf{x}) \ : \ \mathbf{a}_i(\boldsymbol{\xi})^\top\mathbf{x} \leq \frac{b_i(\boldsymbol{\xi})}{1 + \gamma\hat{\sigma}_i} \quad \forall i = 1,\ldots,d \right\}, \tag{8}$$

where $\hat{\sigma}_i$ is empirically estimated from sampled scenarios (see Appendix D.2 for details). Similar to the single-scenario case, we obtain possibly different candidate solutions from different scenarios; whereas parameter γ supplies the conservatism that the single-scenario case lacks and with feasibility defined on the joint constraints rather than through a union bound as in the safe approximation case. Moreover, keeping an estimate per row (constraint) rather than a single estimate across all rows improves solution quality when the rows carry different noise levels, as measured in Appendix E.1.

Third, we avoid excessive conservatism from imposing the scenario constraints by using an additional parameter β to interpolate the constraints between the nominal problem and the realized scenario:

$$\mathbf{a}_i(\boldsymbol{\xi};\beta) = (1-\beta)\,\bar{\mathbf{a}_i} + \beta\,\mathbf{a}_i(\boldsymbol{\xi}), \qquad b_i(\boldsymbol{\xi};\beta) = (1-\beta)\,\bar{b}_i + \beta\,b_i(\boldsymbol{\xi}), \qquad \beta \in [0,1]. \tag{9}$$

Hence, β interpolates between the nominal constraints in equation 7 ($\beta = 0$) and the realized scenario constraints in equation 8 ($\beta = 1$), thereby spanning rather than extending either case. The mixture needs only that $\boldsymbol{\xi}$ can be sampled and requires no further distributional assumptions.

Therefore, the full parameterized subproblem in ACCESS is formulated as

$$\mathbf{x}^\star(\boldsymbol{\xi};\gamma,\beta) \in \arg\max_{\mathbf{x}\in\mathcal{X}} \left\{ f(\mathbf{x}) \;:\; \mathbf{a}_i(\boldsymbol{\xi};\beta)^\top \mathbf{x} \le \frac{b_i(\boldsymbol{\xi};\beta)}{1+\gamma\hat{\sigma}_i} \;\; \forall i \in \{1,\dots,d\} \right\}. \tag{10}$$

In other words, we search over the two-dimensional *parameter space* (γ, β) instead of over the high-dimensional *solution space* $\mathbf{x}$. Parameter γ controls *conservatism*: a larger γ tightens the capacity of the constraints. Parameter β controls *diversity*: a larger β reshapes the constraints towards structurally different solutions. Searching over the parameter space leaves the combinatorial structure to the optimization solver: with γ ordering the subproblems by conservatism and β counterbalancing the spread across scenarios, the search concentrates candidates near the feasibility boundary.

Assumptions and generalizations. The derivation assumed multiplicative independent perturbations and non-negative $\mathbf{a}_i$ and $\mathbf{x}$. These only motivate the parameterization: negative-valued data/decisions and additive rather than multiplicative perturbations need only slight modifications to equation 10, which stays linear in both cases. The β interpolation is natural for continuous distributions, but may require alternative interpolation schemes for discrete distributions, e.g., Bernoulli. The guarantees of Section 3.4 require i.i.d. samples from $\mathbb{P}$ and nothing else, whereas safe approximations need distributional knowledge, e.g., a moment-generating-function bound for Bernstein, first and second moments for Cantelli (Calafiore & El Ghaoui, 2006; Nemirovski & Shapiro, 2006)).

3.2 Structured Sampling via Bayesian Optimization

ACCESS draws N scenarios and runs an independent two-dimensional search for each one. Within a scenario, the search maximizes the subproblem objective subject to maintaining α-feasibility on a set of sample scenarios $\mathcal{V}'$. The set $\mathcal{V}'$ only scores the points within each search, while the N candidates the searches return are scored on a much larger set $\mathcal{V}$ that no search has seen, which is what allows the final selection to be certified (Section 3.4).

Bayesian optimization. For each scenario, we model the $(\gamma, \beta) \to (\text{objective}, \text{feasibility})$ mapping as a constrained black-box problem and solve it with Gaussian process surrogates (Rasmussen & Williams, 2006) using constrained expected improvement (Balandat et al., 2020; Ament et al., 2023), with a small set of quasi-random samples. We search over $\beta \in [0,1]$ and $\gamma \in [0, \gamma_u]$, where γ_u is set per instance by solving the nominal problem over a coarse grid of γ and taking twice the smallest value whose solution is α-feasible on $\mathcal{V}'$. Implementation details and the empirical behaviour of the search appear in Appendices D.1 and D.3–D.5.

3.3 GPU-Accelerated Feasibility Evaluation

Feasibility evaluation decomposes across candidate–scenario pairs: each indicator $\mathbb{I}\{\mathbf{A}(\boldsymbol{\xi}^{(v)})\mathbf{x}^{(i)} \le \mathbf{b}\}$ is independent. We compute all $N \times |\mathcal{V}|$ indicators in a single batched GPU operation, enabling evaluation of $N = 1000$ candidates over $|\mathcal{V}| = 10^6$ scenarios in seconds. This extends naturally to any risk measure that decomposes across scenarios, such as CVaR or variance-based objectives. In two-stage stochastic programs, evaluating a candidate requires solving a recourse problem per scenario, generally precluding batched computation unless the recourse admits specific structure (e.g., dynamic programming representability (Zhao et al., 2026)). Chance constraints require only

Algorithm 1 ACCESS: Adaptive Chance-Constrained Exploration via Structured Sampling

Require: Distribution $\mathbb{P}$, confidence level α, scenarios N, budget $N_{\text{init}} + N_{\text{iter}}$, scenario sets $\mathcal{V}', \mathcal{V}$
Ensure: α-feasible solution $\mathbf{x}^*$

```
Draw scenarios {ξ^(ℓ)}_{ℓ=1}^N ~ ℙ;  C ← ∅
for j = 1, ..., N in parallel do
    for t = 1, ..., N_init + N_iter do
        if t ≤ N_init then
            (γ_t, β_t) ← quasi-random sample from the (γ, β) box
        else
            (γ_t, β_t) ← constrained BO acquisition
        Solve MILP equation 10: x_t ← x*(ξ^(ℓ), γ_t, β_t)        ▷ parametrized subproblems
        Score p̂_{V'}(x_t); update BO
    C ← C ∪ arg max{f(x) : x ∈ {x_t}_{t=1}^{N_init+N_iter}, p̂_{V'}(x) ≥ α}        ▷ best per scenario
x* ← arg max{f(x) : x ∈ C, p̂_V(x) ≥ α}        ▷ p̂_V GPU-batched
return x*
```

indicator evaluation, making our setting trivially GPU-parallel, a key advantage of decoupling that our proposed pipeline leverages.

3.4 Feasibility Guarantees

ACCESS relies on empirical feasibility, which is only an estimate, whereas the true feasibility of the returned solution can fall below estimates on $\mathcal{V}$. Decoupling optimization from evaluation makes this shortfall easy to bound: since ACCESS generates its N candidates without using $\mathcal{V}$, a union bound over those N solutions suffices, and the shortfall is at most a small ϵ_δ with probability at least $1-\delta$, whichever candidate is selected. Raising the selection threshold by ϵ_δ therefore yields ACCESS(δ), whose returned solution is feasible under the true distribution $\mathbb{P}$ with probability at least $1-\delta$.

Proposition 1. *Fix $\delta \in (0,1)$ and let $\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_N$ be generated independently of $\mathcal{V}$. For each j, let $p_j = \Pr_{\boldsymbol{\xi}}(\mathbf{A}(\boldsymbol{\xi})\hat{\mathbf{x}}_j \le \mathbf{b}(\boldsymbol{\xi}))$ be the true feasibility probability of $\hat{\mathbf{x}}_j$ and $\hat{p}_j$ its empirical rate on $\mathcal{V}$. With probability at least $1-\delta$ over the draw of $\mathcal{V}$,*

$$p_j \ge \hat{p}_j - \sqrt{\frac{\ln(N/\delta)}{2|\mathcal{V}|}} \quad \textit{simultaneously for every } j = 1, \ldots, N. \tag{11}$$

The candidates satisfy the condition because the search scores them on $\mathcal{V}'$, which is drawn independently from $\mathcal{V}$. The proof is based on Hoeffding's inequality (Hoeffding, 1963) with a union bound over the N candidates (see Appendix C.2 for details).

Certified selection: ACCESS(δ). Proposition 1 turns the empirical rate into a certificate. Selecting any candidate with

$$\hat{p}_{\mathcal{V}}(\mathbf{x}) \ge \alpha + \epsilon_\delta, \qquad \epsilon_\delta = \sqrt{\frac{\ln(N/\delta)}{2|\mathcal{V}|}}, \tag{12}$$

guarantees $p(\mathbf{x}) \ge \alpha$ with probability at least $1-\delta$. We call this variant ACCESS(δ): the same selection rule with the threshold raised by ϵ_δ.

Comparison with SAA. SAA also offers a similar style of feasibility guarantee. However, it embeds its scenario set in the optimization problem, so its guarantee must hold over every decision in the sample-approximation feasible region, a multiplicity that can be exponential for integer programs (Luedtke & Ahmed, 2008). Correcting over N pre-generated candidates instead lets us evaluate on very large scenarios sets, i.e., $|\mathcal{V}| = 10^6$ scenarios while keeping optimization cheap. For example, with $N = 1000$ and $\delta = 0.05$ we get $\epsilon_\delta = 0.0022$. The guarantee is distribution-free and needs no parametric assumption on $\mathbb{P}$.

4 Computational Results

Computing setup. We used a machine with an NVIDIA H100 GPU and an Intel 6972P CPU with 64GB of RAM. All MILPs were solved on Gurobi 12.0.0 (Gurobi Optimization, LLC, 2024); all neural network implementations and GPU computations use PyTorch 2.5.0 (Paszke et al., 2019).

Benchmarks. We use three problem classes: (1) *Knapsack (KP)*, selecting from n items to maximize profit under d random capacity constraints; (2) *Maximum Weight Matching (MWM)*, a maximum-weight matching on an $n \times n$ bipartite graph under d random edge-weight constraints; and (3) *Resource-Constrained Shortest Path (SRCSP)*, a shortest path through a graph with $|A|$ arcs under d random resource constraints. Each has three sizes (Small, Medium, Large): (n, d) or $(|A|, d) \in \{(20, 3), (50, 5), (100, 10)\}$ for KP and SRCSP, and $n \in \{10, 15, 20\}$ with $d = 5$ for MWM. Each has multiplicative uncertainty, $\mathbf{a}_i(\boldsymbol{\xi}) = \bar{\mathbf{a}}_i \odot \max(\mathbf{1} + \sigma\boldsymbol{\xi},\, 0)$ with $\sigma = 0.2$ and one i.i.d. $\xi_j \sim \mathcal{N}(0, 1)$ per decision variable, shared across the d rows. We then focus on the non-Gaussian case for KP at $n \in \{50, 100\}$, drawing $\boldsymbol{\xi}$ from four distributions: (1) heavy tails (*Student-t_3*), (2) rare large shocks (*Jump*), (3) two correlated blocks of the most attractive items (*Clustered*), and (4) a fixed random 30% of items carrying triple the noise (*Risky subset*). We also explore heterogeneous uncertainty by varying $\boldsymbol{\sigma}$ across constraints in three further configurations (Appendix E.1). For every configuration, we consider 20 instances and confidence levels $\alpha \in \{0.90, 0.95, 0.99\}$, resulting in a total of 1,200 instances. Formulations, distribution parameters, and generation procedures are in Appendix A.

Methods. We evaluate two variants of our approach: ACCESS and ACCESS(δ) with $\delta = 0.05$, which returns a solution that is α-feasible with probability at least $1 - \delta$ over the draw of $\mathcal{V}$. The two differ only in the selection rule and share the same candidate pool, generated from $N = 1000$ scenarios with $N_{\text{init}} = 5$ initialization points and $N_{\text{iter}} = 10$ BO iterations per scenario. We use $|\mathcal{V}'| = 10^4$ scenarios to score points within the BO and $|\mathcal{V}| = 10^6$ for final selection. Every subproblem solve is capped at 3 s on a single thread, although most solves terminate in orders of magnitude less time, and we parallelize across scenarios with 40 workers. Reported runtimes cover both phases, the CPU subproblem solves and the GPU feasibility evaluation over $\mathcal{V}$, and Appendix I reports both separately.

Baselines. We compare against (1) the big-M sample average approximation (SAA) (Luedtke & Ahmed, 2008), (2) the sample CVaR approximation applied per row with a Bonferroni split (Rockafellar & Uryasev, 2000; Nemirovski & Shapiro, 2006), and (3) the Gaussian–Bonferroni (GB) safe approximation, which is exact for each row under Gaussian $\boldsymbol{\xi}$ (Lobo et al., 1998, Section 2.6) and conservative for the joint constraint through the Bonferroni split. GB does not apply to the non-Gaussian benchmarks and is omitted there. The main results use $|\mathcal{S}| = 1000$ scenarios for SAA and CVaR, drawn independently of $\mathcal{V}$ and $\mathcal{T}$; results for $|\mathcal{S}| \in \{10, 100, 10000\}$ are in Appendix G. All baseline solves use Gurobi with 6 threads and a 3600 s time limit, returning the best incumbent at the limit. Baseline formulations and details are in Appendix B.

Evaluation metrics. We evaluate feasibility using $\mathcal{T} = 10^6$ sampled scenarios per instance via equation 4 (separate from the selection set $\mathcal{V}$ used in ACCESS). We report runtime, α-feasibility rate (fraction of instances where the method finds an α-feasible solution), and two gap metrics. A per-benchmark runtime breakdown is in Appendix I. *FO-gap* (Table 1) measures the gap to the Best Known Objective (BKO). On each instance, FO_m is the objective value of the solution method m returns, and BKO is the largest FO_m over the methods whose returned solution is α-feasible on $\mathcal{T}$. FO-gap$_m = (\text{BKO} - \text{FO}_m)/\text{BKO}$ is reported only when the solution of m is itself α-feasible, and averaged over those instances. *Optimality gap* (Figure 3) uses the same reference as FO-gap, the best α-feasible objective found by any method, so figures and tables agree; every solution is plotted, feasible or not, and an infeasible solution that beats the reference appears at a negative gap. This shows the full feasibility-quality tradeoff without a second definition of the gap.

Results. Table 1 summarizes results across all benchmarks with Gaussian uncertainty. ACCESS attains the best or tied-best FO-gap in 26 of the 27 settings, within 0.04% of the best known objective throughout. ACCESS falls below α on a small minority of instances; ACCESS(δ) is α-feasible on every instance, at a worst case of 0.64%. SAA's gaps are averaged only over the instances where it returns a feasible solution, a subset that falls to 0% on the two larger KP sizes at α=0.99, so the comparison flatters it. The safe approximations invert the trade-off: GB is α-feasible on every instance and CVaR on all but three settings, but they give up 1.6–7.5% and 0.9–10.4% of objective respectively. Per-instance distributions are in Appendix H.

Figure 3 evaluates all methods when $\boldsymbol{\xi}$ is not Gaussian, on KP at n=100 and α=0.99. The three methods separate the same way in every panel. ACCESS is α-feasible on nearly all instances with mean gap of at most 0.32%. CVaR is α-feasible on all 80 instances, but pays in solution quality. SAA stays below α on 65% of instances even at $|\mathcal{S}|$=10^4, and its negative gaps are solutions that beat the best α-feasible objective while violating the chance constraint. See Appendix E.2 for details.

Table 1: Feasibility, objective, and runtime results. Each cell reports FO-gap / α-feasibility rate, followed by runtime in seconds (averaged over α values). FO-gap is relative to BKO (Best Known Objective): the best α-feasible objective across all methods per instance. SAA and CVaR use $|\mathcal{S}|$=1000 scenarios; ACCESS(δ) applies Hoeffding correction with δ=0.05. Sizes: KP: $n \in \{20, 50, 100\}$, $d \in \{3, 5, 10\}$; MWM: $10\times10/15\times15/20\times20$, $d = 5$; SRCSP: $|A| \in \{20, 50, 100\}$, $d \in \{3, 5, 10\}$. Each entry is averaged over 20 test instances.

		Small				Medium				Large			
	Method	Time	α=.90	α=.95	α=.99	Time	α=.90	α=.95	α=.99	Time	α=.90	α=.95	α=.99
KP	SAA	574	0.08% / 75%	0.11% / 50%	**0.00%** / 45%	3588	0.21% / 40%	0.19% / 25%	— / 0%	3600	0.11% / 25%	0.29% / 5%	— / 0%
	CVaR	2.5	5.82% / 100%	4.88% / 100%	3.33% / 100%	7.3	4.16% / 100%	3.40% / 100%	1.37% / 85%	156	3.25% / 100%	2.45% / 100%	0.89% / 35%
	GB	0.1	3.52% / 100%	2.94% / 100%	2.54% / 100%	1.5	3.00% / 100%	2.65% / 100%	2.14% / 100%	111	2.87% / 100%	2.51% / 100%	2.05% / 100%
	ACCESS	88	**0.00%** / 100%	**0.01%** / 100%	0.00% / 100%	111	**0.00%** / 95%	**0.04%** / 100%	**0.00%** / 100%	129	**0.01%** / 100%	**0.01%** / 90%	**0.00%** / 95%
	ACCESS(δ)	88	0.07% / 100%	0.10% / 100%	0.50% / 100%	111	0.06% / 100%	0.11% / 100%	0.39% / 100%	129	0.03% / 100%	0.05% / 100%	0.27% / 100%
MWM	SAA	1943	0.09% / 75%	0.17% / 75%	0.13% / 55%	3600	0.80% / 85%	0.62% / 55%	0.11% / 55%	3600	1.38% / 65%	0.84% / 80%	0.04% / 40%
	CVaR	14	9.20% / 100%	6.76% / 100%	5.15% / 100%	286	5.75% / 100%	5.18% / 100%	3.39% / 95%	2372	4.49% / 100%	3.85% / 100%	1.94% / 100%
	GB	1.9	5.71% / 100%	4.58% / 100%	4.07% / 100%	39	3.62% / 100%	3.04% / 100%	2.63% / 100%	761	2.68% / 100%	2.22% / 100%	1.63% / 100%
	ACCESS	136	**0.00%** / 100%	**0.00%** / 100%	**0.00%** / 100%	291	**0.00%** / 100%	**0.00%** / 95%	**0.00%** / 100%	565	**0.00%** / 100%	**0.00%** / 100%	**0.00%** / 95%
	ACCESS(δ)	136	0.00% / 100%	0.03% / 100%	0.11% / 100%	291	0.09% / 100%	0.06% / 100%	0.56% / 100%	565	0.11% / 100%	0.06% / 100%	0.45% / 100%
SRCSP	SAA	1.1	1.84% / 95%	**0.00%** / 100%	**0.00%** / 95%	21	**0.00%** / 90%	0.98% / 95%	0.79% / 70%	551	0.07% / 85%	0.28% / 75%	0.41% / 55%
	CVaR	0.5	10.38% / 100%	7.16% / 100%	3.10% / 100%	2.9	8.56% / 100%	6.84% / 100%	6.95% / 100%	11	8.62% / 100%	7.44% / 100%	4.62% / 100%
	GB	0.0	7.51% / 100%	3.29% / 100%	2.05% / 100%	0.1	5.77% / 100%	4.98% / 100%	4.30% / 100%	0.5	6.77% / 100%	6.25% / 100%	5.40% / 100%
	ACCESS	81	**0.00%** / 100%	0.00% / 100%	0.00% / 100%	91	0.03% / 100%	**0.00%** / 100%	**0.00%** / 100%	109	**0.04%** / 95%	**0.00%** / 100%	**0.00%** / 100%
	ACCESS(δ)	81	0.00% / 100%	0.00% / 100%	0.00% / 100%	91	0.03% / 100%	0.24% / 100%	0.59% / 100%	109	0.10% / 100%	0.13% / 100%	0.64% / 100%

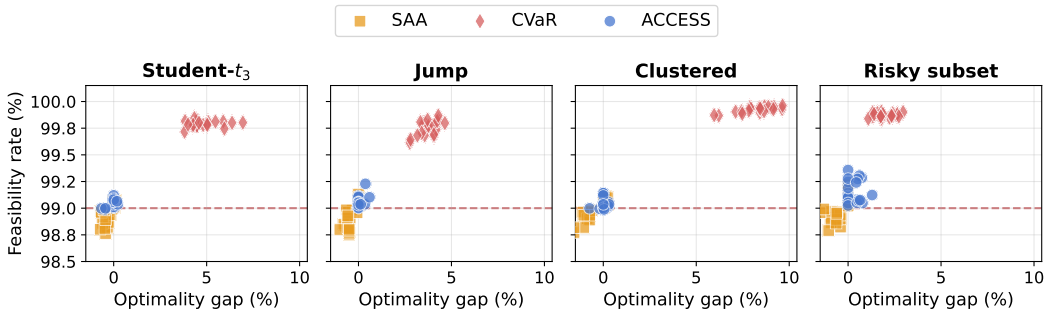

Figure 3: Feasibility rate vs. optimality gap under non-Gaussian distributions, on KP with n=100, α=0.99. Each point is one of 20 instances, and the dashed line marks α. SAA and CVaR use $|\mathcal{S}|$=10^4. Gap is relative to the best α-feasible objective, so infeasible solutions may have negative gaps.

Limitations. ACCESS replaces one large solve by thousands of small independent ones, so its cost presumes a cheap deterministic subproblem and benefits heavily from parallel hardware. Its guarantee concerns feasibility rather than objective value, although empirically the returned solutions lie within a fraction of a percent of the best objective found by any method.

5 Conclusion

We introduced ACCESS, a structured sampling approach for the computationally efficient and effective solution of chance-constrained stochastic programs by reducing the search from high-dimensional solution space to a 2D (γ, β) parameter space while capturing the trade-off between optimality and feasibility. By concentrating candidates near the feasibility boundary via Bayesian optimization, ACCESS consistently finds high-quality α-feasible solutions while ACCESS(δ) provides certified guarantees with probability $1 - \delta$. Empirically, ACCESS attains near-zero optimality gap at high feasibility across all benchmarks, where SAA is frequently infeasible and the safe approximations are uniformly conservative. Future directions include extending to alternative probabilistic constraint structures and risk measures such as distributionally robust formulations, generalizing to richer formulations that allow for more flexibility, and investigating alternative search strategies beyond Bayesian optimization.

AI USE STATEMENT

We used generative AI tools for feedback on the design of the method and experiments, to implement parts of the code, and to summarize experimental output. We did not use these tools to formulate or prove mathematical claims or to generate data, and the remaining required-disclosure tasks do not apply to this work. We also used them to draft and polish parts of the text, most of which the authors then reviewed and rewrote. Every benchmark instance is either adopted from the literature, mostly from extending classic deterministic problems to CCSP problems. We reviewed the AI-assisted work: the code is covered by the released test suite, we regenerated every reported number from the released result files, and the authors checked the text. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

Chance-constrained optimization is widely used in applications requiring reliable decision-making under uncertainty, including logistics, energy systems, transportation, and resource allocation. By improving the scalability of chance-constrained combinatorial optimization, our work may help support planning systems that better account for uncertainty and rare events. At the same time, these methods could also be applied in domains involving strategic or security-sensitive decision-making. As with many optimization technologies, the broader societal impact depends on the application context and deployment choices.

REPRODUCIBILITY STATEMENT

All code, trained models, result files, and instructions are available in an anonymous repository at `https://anonymous.4open.science/r/access-5460/README.md`. Appendix A lists the benchmarks, and Appendix D.1 lists the parameters.

REFERENCES

Shabbir Ahmed and Dimitri J Papageorgiou. Probabilistic set covering with correlations. *Operations Research*, 61(2):438–452, 2013.

Shabbir Ahmed and Alexander Shapiro. Solving chance-constrained stochastic programs via sampling and integer programming. In *State-of-the-art decision-making tools in the information-intensive age*, pp. 261–269. Informs, 2008.

Antonio Alcántara and Carlos Ruiz. On data-driven chance constraint learning for mixed-integer optimization problems. *Applied Mathematical Modelling*, 121:445–462, 2023.

Sebastian Ament, Sam Daulton, David Eriksson, Maximilian Balandat, and Eytan Bakshy. Unexpected improvements to expected improvement for bayesian optimization. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL `https://openreview.net/forum?id=1vyAG6j9PE`.

Maximilian Balandat, Brian Karrer, Daniel R. Jiang, Samuel Daulton, Benjamin Letham, Andrew Gordon Wilson, and Eytan Bakshy. Botorch: A framework for efficient monte-carlo bayesian optimization. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust Optimization*. Princeton University Press, 2009.

Dimitris Bertsimas and Melvyn Sim. The price of robustness. *Operations Research*, 52(1):35–53, 2004.

Giuseppe C. Calafiore and Laurent El Ghaoui. On distributionally robust chance-constrained linear programs. *Journal of Optimization Theory and Applications*, 130(1):1–22, 2006.

Wenqing Chen, Melvyn Sim, Jie Sun, and Chung-Piaw Teo. From CVaR to uncertainty set: Implications in joint chance-constrained optimization. *Operations Research*, 58(2):470–485, 2010.

Xin Chen, Melvyn Sim, and Peng Sun. A robust optimization perspective on stochastic programming. *Operations Research*, 55(6):1058–1071, 2007.

Justin Dumouchelle, Esther Julien, Jannis Kurtz, and Elias B Khalil. Neur2bilo: Neural bilevel optimization. *Advances in Neural Information Processing Systems*, 37:86688–86719, 2024.

Maxime Gasse, Didier Chételat, Nicola Ferroni, Laurent Charlin, and Andrea Lodi. Exact combinatorial optimization with graph convolutional neural networks. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Gurobi Optimization, LLC. *Gurobi Optimizer Reference Manual*, 2024. URL `https://www.gurobi.com`.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

Yuma Ichikawa and Yamato Arai. Optimization by parallel quasi-quantum annealing with gradient-based sampling. In *The Thirteenth International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=9EfBeXaXf0`.

Stefan Irnich and Guy Desaulniers. Shortest path problems with resource constraints. In Guy Desaulniers, Jacques Desrosiers, and Marius M. Solomon (eds.), *Column Generation*, pp. 33–65. Springer, 2005.

Elias B. Khalil, Pierre Le Bodic, Le Song, George Nemhauser, and Bistra Dilkina. Learning to branch in mixed integer programming. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

Wouter Kool, Herke van Hoof, and Max Welling. Attention, learn to solve routing problems! In *International Conference on Learning Representations*, 2019.

Jan Kronqvist, Boda Li, Jan Rolfes, and Shudian Zhao. Alternating mixed-integer programming and neural network training for approximating stochastic two-stage problems. In *International Conference on Machine Learning, Optimization, and Data Science*, pp. 124–139. Springer, 2023.

Simge Küçükyavuz and Ruiwei Jiang. Chance-constrained optimization under limited distributional information: A review of reformulations based on sampling and distributional robustness. *EURO Journal on Computational Optimization*, 10:100030, 2022.

Zhuangzhi Li and Zukui Li. Optimal robust optimization approximation for chance constrained optimization problem. *Computers & Chemical Engineering*, 74:89–99, 2015.

Miguel Sousa Lobo, Lieven Vandenberghe, Stephen Boyd, and Hervé Lebret. Applications of second-order cone programming. *Linear Algebra and its Applications*, 284(1–3):193–228, 1998.

James Luedtke and Shabbir Ahmed. A sample approximation approach for optimization with probabilistic constraints. *SIAM Journal on Optimization*, 19(2):674–699, 2008.

James Luedtke, Shabbir Ahmed, and George L Nemhauser. An integer programming approach for linear programs with probabilistic constraints. *Mathematical programming*, 122(2):247–272, 2010.

Donato Maragno, Holly Wiberg, Dimitris Bertsimas, Ş İlker Birbil, Dick den Hertog, and Adejuyigbe O Fajemisin. Mixed-integer optimization with constraint learning. *Operations Research*, 73(2):1011–1028, 2025.

Arkadi Nemirovski. On safe tractable approximations of chance constraints. *European Journal of Operational Research*, 219(3):707–718, 2012.

Arkadi Nemirovski and Alexander Shapiro. Convex approximations of chance constrained programs. *SIAM Journal on Optimization*, 17(4):969–996, 2006.

Bernardo K Pagnoncelli, Shabbir Ahmed, and Alexander Shapiro. Sample average approximation method for chance constrained programming: theory and applications. *Journal of Optimization Theory and Applications*, 142(2):399–416, 2009.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Rahul Mihir Patel, Justin Dumouchelle, Elias Khalil, and Merve Bodur. Neur2sp: Neural two-stage stochastic programming. *Advances in neural information processing systems*, 35:23992–24005, 2022.

David Pisinger. Where are the hard knapsack problems? *Computers & Operations Research*, 32(9): 2271–2284, 2005.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

R. Tyrrell Rockafellar and Stanislav Uryasev. Optimization of conditional value-at-risk. *Journal of Risk*, 2:21–41, 2000.

Alexander Shapiro, Darinka Dentcheva, and Andrzej Ruszczyński. *Lectures on Stochastic Programming: Modeling and Theory*. SIAM, 2009.

Haoran Sun, Katayoon Goshvadi, Azade Nova, Dale Schuurmans, and Hanjun Dai. Revisiting sampling for combinatorial optimization. *International Conference on Machine Learning*, 2023.

Zhiqing Sun and Yiming Yang. Difusco: Graph-based diffusion solvers for combinatorial optimization. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Yuan Yuan, Zukui Li, and Biao Huang. Robust optimization approximation for joint chance constrained optimization problem. *Journal of Global Optimization*, 67(4):805–827, 2017. doi: 10.1007/s10898-016-0438-0.

Jingyi Zhao, Linxin Yang, Haohua Zhang, Qile He, and Tian Ding. From sequential to parallel: Reformulating dynamic programming as gpu kernels for large-scale stochastic combinatorial optimization. In *International Conference on Learning Representations*, 2026.

Steve Zymler, Daniel Kuhn, and Berç Rustem. Distributionally robust joint chance constraints with second-order moment information. *Mathematical Programming*, 137(1–2):167–198, 2013.

Appendix

A Benchmarks and Details 13
B Baselines 15
C Certified Feasibility Guarantees 17
D Algorithmic Details 19
E Non-Standard Uncertainty 23
F Ablation: Nominal-Only Tightening 26
G Baseline Analysis 27
H Per-Instance Distributional Results 28
I Detailed Runtime Results 37

A BENCHMARKS AND DETAILS

Three benchmarks share the same uncertainty model. Every random coefficient is perturbed multiplicatively, $a(\boldsymbol{\xi}) = \bar{a} \odot \max(1 + \sigma\xi, 0)$ with $\xi \sim \mathcal{N}(0, 1)$, where the truncation keeps the coefficient non-negative. Perturbations of this form are common in the CCSP literature (Luedtke & Ahmed, 2008; Ahmed & Shapiro, 2008; Nemirovski & Shapiro, 2006). We use $\sigma = 0.2$ on every constraint unless stated otherwise, and we evaluate methods on each benchmark at three sizes and three confidence levels $\alpha \in \{0.90, 0.95, 0.99\}$ over 20 held-out instances per setting, which gives 540 instances under Gaussian $\boldsymbol{\xi}$. Two further settings reuse the knapsack generator and change only the uncertainty, (1) the heterogeneous σ_d of Appendix E.1 and (2) the four non-Gaussian distributions of Appendix E.2.

A.1 BINARY CHANCE-CONSTRAINED KNAPSACK

The multi-dimensional binary chance-constrained knapsack with n items and d resource dimensions is given as

$$\max_{x \in \{0,1\}^n} \quad \sum_{j=1}^{n} v_j x_j \tag{13a}$$

$$\text{s.t.} \quad \Pr_{\xi}\left(\sum_{j=1}^{n} \tilde{w}_{i,j}(\boldsymbol{\xi})\, x_j \le b_i,\ \forall\, i = 1, \dots, d \right) \ge \alpha. \tag{13b}$$

where x_j selects item j, $v_j > 0$ is its value, $w_{i,j} > 0$ its nominal weight in dimension i, and b_i the deterministic capacity of dimension i. A scenario carries one ξ_j per item, so the perturbed weight is $\tilde{w}_{i,j}(\boldsymbol{\xi}) = w_{i,j} \max(1 + \sigma\xi_j,\, 0)$ with $\boldsymbol{\xi} \sim \mathcal{N}(0, I_n)$.

Weights are drawn as $w_{i,j} \sim \text{Uniform}\{1, \dots, 100\}$ and capacities as $b_i = \rho \sum_j w_{i,j}$ with $\rho = 0.5$. Values follow the weakly correlated type of Pisinger (2005), $v_j = \max(w_{1,j} + \epsilon,\, 1)$ with $\epsilon \sim \text{Uniform}\{-10, \dots, 10\}$. The three sizes are n=20, d=3; n=50, d=5; and n=100, d=10.

A.2 MAXIMUM WEIGHT BIPARTITE MATCHING

The chance-constrained maximum weight bipartite matching with d resource budgets is given as

$$\max_{x \in \{0,1\}^{mn}} \quad \sum_{i,j} p_{ij}\, x_{ij} \tag{14a}$$

$$\text{s.t.} \quad \Pr_{\xi}\left(\sum_{i,j} \tilde{r}_{k,ij}(\boldsymbol{\xi})\, x_{ij} \le B_k,\ \forall\, k = 1, \dots, d \right) \ge \alpha, \tag{14b}$$

$$\sum_{j} x_{ij} \le 1 \quad \forall\, i \in U, \qquad \sum_{i} x_{ij} \le 1 \quad \forall\, j \in V, \tag{14c}$$

where x_{ij} selects edge (i, j) of a complete bipartite graph $G = (U, V, E)$ with $|U| = m$ and $|V| = n$, p_{ij} is the edge profit, $r_{k,ij}$ its nominal consumption of resource k, and B_k the budget of resource k. A scenario carries one ξ_{ij} per edge, so $\tilde{r}_{k,ij}(\boldsymbol{\xi}) = r_{k,ij} \max(1 + \sigma\xi_{ij}, 0)$ with $\xi_{ij} \sim \mathcal{N}(0, 1)$.

Resources are drawn as $r_{k,ij} \sim \text{Uniform}\{1, \dots, 10\}$ and profits as $p_{ij} = \max(\bar{r}_{ij} + \epsilon,\, 1)$ with $\epsilon \sim \text{Uniform}\{-2, \dots, 2\}$, where $\bar{r}_{ij} = d^{-1} \sum_k r_{k,ij}$. Budgets are $B_k = \rho \min(m, n) \bar{r}_k$ with $\rho = 0.5$, where $\bar{r}_k$ is the mean nominal consumption of resource k and $\min(m, n)$ is the largest number of edges in a matching. The three sizes are m=n=10, 15 and 20 at d=5, giving 100, 225 and 400 variables.

A.3 Stochastic Resource-Constrained Shortest Path

The resource-constrained shortest path (Irnich & Desaulniers, 2005) with stochastic arc consumptions is given as

$$\max_{x\in\{0,1\}^{|A|}} \quad \sum_{e\in A} c_e\, x_e \tag{15a}$$

$$\text{s.t.} \quad \Pr_\xi\left(\sum_{e\in A} \tilde{r}_{i,e}(\boldsymbol{\xi})\, x_e \le B_i,\ \forall\, i=1,\dots,d\right) \ge \alpha, \tag{15b}$$

$$\sum_{e:\,\mathrm{tail}(e)=v} x_e - \sum_{e:\,\mathrm{head}(e)=v} x_e = b_v \quad \forall\, v\in V, \tag{15c}$$

where x_e selects arc e of a directed acyclic graph $G=(V,A)$, $c_e>0$ is the arc profit, $r_{i,e}>0$ its nominal consumption of resource i, and B_i the budget of resource i. The flow-conservation right-hand side is $b_s=1$, $b_t=-1$, and $b_v=0$ at every internal node. A scenario carries one ξ_e per arc, so $\tilde{r}_{i,e}(\boldsymbol{\xi}) = r_{i,e}\max(1+\sigma\xi_e, 0)$ with $\xi_e \overset{\text{iid}}{\sim} \mathcal{N}(0,1)$.

Each graph is a sequential backbone $0\to 1\to\cdots\to(|V|-1)$, which guarantees an s-t path, plus random skip arcs (i,j) with $j\ge i+2$ up to exactly $|A|$ arcs. Arc resources are drawn as $r_{i,e}\sim\mathrm{Uniform}\{1,\dots,100\}$ and profits as $c_e=\max(r_{1,e}+\epsilon,\,1)$ with $\epsilon\sim\mathrm{Uniform}\{-10,\dots,10\}$. Budgets are $B_i=\rho\,(|V|-1)\,\bar{r}_i$ with $\rho=0.65$, where $\bar{r}_i$ is the mean nominal resource in dimension i and $|V|-1$ is the largest number of arcs on a path. The three sizes are $|V|$=10, $|A|$=20, d=3; $|V|$=14, $|A|$=50, d=5; and $|V|$=18, $|A|$=100, d=10.

B BASELINES

Each baseline solves one deterministic surrogate of equation 1 and returns a single solution. SAA and CVaR embed $|\mathcal{S}|$ i.i.d. scenarios $\mathcal{S} = \{\boldsymbol{\xi}^{(s)}\}_{s=1}^{|\mathcal{S}|}$ drawn from $\mathbb{P}$ through a set independent of the test set $\mathcal{T}$, and the Gaussian–Bonferroni (GB) approximation uses the model σ without requiring scenarios. All three are solved with Gurobi at 6 threads and a 3600 s limit. Each solution is then evaluated on $\mathcal{T}$ exactly as the candidate ACCESS selects is, and counts as α-feasible only if its empirical feasibility on $\mathcal{T}$ is at least α.

B.1 SAA FORMULATION

The sample average approximation (SAA) replaces the chance constraint of equation 1 by its empirical counterpart over $\mathcal{S}$ and linearises it with one indicator per scenario (Luedtke & Ahmed, 2008):

$$\max_{\mathbf{x}\in\mathcal{X}} \Big\{ f(\mathbf{x}) : \mathbf{A}(\boldsymbol{\xi}^{(s)})\mathbf{x} \le \mathbf{b}(\boldsymbol{\xi}^{(s)}) + \mathbf{M}(1-y_s) \;\; \forall s = 1,\dots,|\mathcal{S}|, \;\; \sum_{s=1}^{|\mathcal{S}|} y_s \ge \lceil \alpha|\mathcal{S}| \rceil, \;\; \mathbf{y} \in \{0,1\}^{|\mathcal{S}|} \Big\}, \tag{16}$$

where $y_s = 1$ marks a scenario whose constraints are enforced and $\mathbf{M} \in \mathbb{R}^d_+$ is a vector of big-M constants, one per row. The cardinality constraint allows at most $\lfloor (1-\alpha)|\mathcal{S}| \rfloor$ sampled scenarios to be violated, so equation 16 is the chance constraint at risk level $1-\alpha$ under the empirical distribution of $\mathcal{S}$. We use the plain big-M model without the strengthening inequalities of Luedtke et al. (2010). We set $\mathbf{M}$ from the sampled scenarios with $\hat{a}_{ij} = \max_s a_{ij}(\boldsymbol{\xi}^{(s)})$ the largest sampled coefficient and $\bar{b}_i$ the right-hand side, which is deterministic in all three families,

$$M_i = \begin{cases} \sum_{j=1}^{n} \hat{a}_{ij} - \bar{b}_i & \text{knapsack, where any subset of items may be selected,} \\ \sum_{j\in T_i} \hat{a}_{ij} - \bar{b}_i & \text{matching, with } T_i \text{ the } \min(|U|,|V|) \text{ largest } \hat{a}_{ij}, \\ (|V|-1)\,\max_j \hat{a}_{ij} - \bar{b}_i & \text{shortest path, where a path uses at most } |V|-1 \text{ arcs,} \end{cases}$$

each floored at 10^{-3}, so M_i dominates the largest violation any $\mathbf{x} \in \mathcal{X}$ can have on $\mathcal{S}$.

B.2 SAFE APPROXIMATIONS

Safe approximations replace the chance constraint equation 1b by a deterministic convex constraint that implies it, so every feasible point of the surrogate is α-feasible by construction (Nemirovski & Shapiro, 2006; Chen et al., 2007; Nemirovski, 2012). For a single row whose usage is a linear combination of independent zero-mean perturbations, the classical members of this family share the mean-plus-margin form of equation 17 and differ only in the constant κ and in what they assume about $\boldsymbol{\xi}$. Each bounds an individual constraint, so a joint constraint over d rows is handled either by splitting the risk across rows or by a joint distributionally robust reformulation (Chen et al., 2010; Zymler et al., 2013), whereas SAA enforces the joint constraint on the sampled scenarios directly. We run the two members that apply to our benchmarks, the Gaussian reformulation with a Bonferroni split and the sample CVaR.

Bonferroni approximation. For a joint chance constraint over d rows, the Bonferroni inequality requires each row i to hold with probability at least α_i with $\sum_{i=1}^{d}(1-\alpha_i) \le 1-\alpha$, and we use the uniform allocation $\alpha_i = 1-(1-\alpha)/d$. The bound ignores dependence between rows. In our benchmarks every row is driven by the same item-level noise $\boldsymbol{\xi}$, so the rows are strongly positively dependent and the bound is loose.

Gaussian–Bonferroni (GB) approximation. The GB baseline imposes the joint constraint equation 1b one row at a time. Here we take a single row at level α, $\Pr(\mathbf{a}_i(\boldsymbol{\xi})^\top \mathbf{x} \le \bar{b}_i) \ge \alpha$. For the multiplicative Gaussian model, which a subset of our benchmarks uses, $\mathbf{a}_i(\boldsymbol{\xi}) = \bar{\mathbf{a}}_i \odot (1 + \sigma_i \boldsymbol{\xi})$ with $\boldsymbol{\xi} \sim \mathcal{N}(0, \mathbf{I})$, we have $\mathbf{a}_i(\boldsymbol{\xi})^\top \mathbf{x} \sim \mathcal{N}\big(\bar{\mathbf{a}}_i^\top \mathbf{x},\, \sigma_i^2 \|\bar{\mathbf{a}}_i \odot \mathbf{x}\|_2^2\big)$, so for $\alpha \ge 1/2$ this is equivalent to the second-order cone constraint

$$\bar{\mathbf{a}}_i^\top \mathbf{x} + \Phi^{-1}(\alpha)\, \sigma_i \|\bar{\mathbf{a}}_i \odot \mathbf{x}\|_2 \le \bar{b}_i, \tag{17}$$

where $\Phi^{-1}(\cdot)$ is the inverse standard normal CDF (Lobo et al., 1998, Section 2.6). This is the exact form of the individual constraint, so GB is the tightest member of the family under the Gaussian model.

Imposing equation 17 on every row with α replaced by the Bonferroni level $\alpha_i = 1 - (1-\alpha)/d$ then guarantees equation 1b. When $\mathbf{x}$ is binary, the result is a mixed-integer second-order cone program. The benchmarks clip the coefficients at zero, so equation 17 is exact for the untruncated model only. However, we note that with $\sigma_i = 0.2$, truncation among sampled scenarios is exceedingly rare.

CVaR approximation. Conditional value-at-risk gives the tightest convex conservative approximation of an individual chance constraint (Nemirovski & Shapiro, 2006; Rockafellar & Uryasev, 2000): for the violation $g_i(\mathbf{x}, \boldsymbol{\xi}) = \mathbf{A}(\boldsymbol{\xi})_i \mathbf{x} - b_i(\boldsymbol{\xi})$ of row i and a budget $\varepsilon_i \in (0, 1)$,

$$\mathrm{CVaR}_{\varepsilon_i}\big(g_i(\mathbf{x}, \boldsymbol{\xi})\big) = \min_{t \in \mathbb{R}} \Big\{ t + \frac{1}{\varepsilon_i} \mathbb{E}\big[(g_i(\mathbf{x}, \boldsymbol{\xi}) - t)^+\big] \Big\} \leq 0 \tag{18}$$

implies $\Pr(g_i(\mathbf{x}, \boldsymbol{\xi}) \leq 0) \geq 1 - \varepsilon_i$, the CVaR being the mean of the worst ε_i fraction of outcomes. We impose equation 18 row by row with the Bonferroni budgets $\varepsilon_i = (1-\alpha)/d$ and replace the expectation by its average over $\mathcal{S}$. With the auxiliary variables of Rockafellar & Uryasev (2000) the constraints are linear,

$$t_i + \frac{1}{\varepsilon_i |\mathcal{S}|} \sum_{s=1}^{|\mathcal{S}|} w_{s,i} \leq 0, \qquad w_{s,i} \geq \mathbf{A}(\boldsymbol{\xi}^{(s)})_i \mathbf{x} - b_i(\boldsymbol{\xi}^{(s)}) - t_i, \qquad w_{s,i} \geq 0, \qquad i = 1, \ldots, d, \tag{19}$$

adding d thresholds and $d\,|\mathcal{S}|$ continuous variables to the MILP over $\mathbf{x}$, and no binaries. The sample CVaR is conservative only once $\varepsilon_i |\mathcal{S}|$ is large, since when $\varepsilon_i |\mathcal{S}| \leq 1$ it is the largest sampled violation of the row and carries no guarantee. CVaR needs no distributional assumption, so it is the safe approximation we run on the non-Gaussian distributions.

C CERTIFIED FEASIBILITY GUARANTEES

ACCESS(δ) selects a candidate whose true feasibility admits a high-probability lower bound. This section states the bound and the assumptions it needs.

C.1 SETUP

Let N candidate solutions $\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_N$ be generated by ACCESS *without* using the evaluation set $\mathcal{V}$. The candidates are then compared on a single evaluation set $\mathcal{V} = \{\xi^{(1)}, \ldots, \xi^{(|\mathcal{V}|)}\}$ of $|\mathcal{V}|$ i.i.d. draws from the true (unknown) distribution $\mathbb{P}$, and the best feasible-looking one is selected. In our experiments, $N = 1000$ candidates and $|\mathcal{V}| = 10^6$ scenarios. Selecting on $\mathcal{V}$ biases the winner, since the candidate that looks best is the one most likely to have drawn favourably, so the empirical rate needs a multiple-comparisons correction over the N candidates.

For a candidate $\hat{\mathbf{x}}_j$, let

$$p_j := \Pr_{\xi}\big(\mathbf{A}(\xi)\hat{\mathbf{x}}_j \le \mathbf{b}(\xi)\big)$$

be its true (unknown) feasibility probability, and

$$\hat{p}_j := \frac{1}{|\mathcal{V}|}\sum_{i=1}^{|\mathcal{V}|} \mathbb{I}\big\{\mathbf{A}(\xi^{(i)})\hat{\mathbf{x}}_j \le \mathbf{b}(\xi^{(i)})\big\}$$

its empirical rate on $\mathcal{V}$.

C.2 PROOF OF PROPOSITION 1

We restate the proposition of Section 3.4 for reference.

Proposition 1 (restated). *Fix $\delta \in (0,1)$ and let $\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_N$ be generated independently of $\mathcal{V}$. For each j, let $p_j = \Pr_{\boldsymbol{\xi}}(\mathbf{A}(\boldsymbol{\xi})\hat{\mathbf{x}}_j \le \mathbf{b}(\boldsymbol{\xi}))$ be the true feasibility probability of $\hat{\mathbf{x}}_j$ and $\hat{p}_j$ its empirical rate on $\mathcal{V}$. With probability at least $1-\delta$ over the draw of $\mathcal{V}$,*

$$p_j \ge \hat{p}_j - \sqrt{\frac{\ln(N/\delta)}{2|\mathcal{V}|}} \quad \textit{simultaneously for every } j = 1, \ldots, N.$$

Proof. The candidate solutions $\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_N$ are generated without using $\mathcal{V}$. Thus, for any realized candidate pool and each candidate j, the $|\mathcal{V}|$ feasibility indicators evaluated on $\mathcal{V}$ are i.i.d. Bernoulli(p_j).

The proposition's bound $p_j \ge \hat{p}_j - t$ fails exactly when $\hat{p}_j - p_j > t$, i.e., when the empirical rate overshoots the true rate by more than t. By Hoeffding's inequality (Hoeffding, 1963), for any $t > 0$:

$$\Pr\big(\hat{p}_j - p_j \ge t\big) \le e^{-2|\mathcal{V}|t^2}.$$

Taking $t = \sqrt{\ln(N/\delta)/(2|\mathcal{V}|)}$ makes the right-hand side equal δ/N, so this failure event occurs with probability at most δ/N.

By Boole's inequality (union bound), the probability that the bound fails for *any* candidate is:

$$\Pr\big(\exists j : \text{bound fails for } j\big) \le \sum_{j=1}^{N} \Pr\big(\text{bound fails for } j\big) \le \sum_{j=1}^{N} \frac{\delta}{N} = \delta.$$

So the bound holds for all N candidates simultaneously with probability at least $1-\delta$. The event does not reference which j is selected, so it covers whichever index the selection rule returns. □

Certified selection rule. To certify that the selected solution has true feasibility $p_j \ge \alpha$ with probability at least $1-\delta$, we select only candidates satisfying:

$$\hat{p}_j \ge \alpha + \epsilon, \quad \text{where } \epsilon = \sqrt{\frac{\ln(N/\delta)}{2|\mathcal{V}|}}.$$

C.3 Relation to SAA Theory

Proposition 1 is a finite-candidate specialization of the feasibility argument in Luedtke & Ahmed (2008). Their union bound runs over every decision in the sample-approximation feasible region that is truly infeasible but might look empirically feasible, which gives a multiplicity factor based on the size of the feasible region, which in general is exponential for a bounded integer program. They note that the resulting sample sizes are far too large to solve. ACCESS generates its candidates without $\mathcal{V}$, so only those N can be selected and the correction runs over that finite set instead. It needs i.i.d. draws for $\mathcal{V}$ and a pool generated without it, and no distributional assumptions.

C.4 Practical Implications

For our experiments with $|\mathcal{V}| = 10^6$, $N = 1000$, and $\delta = 0.05$, the required margin is:

$$\epsilon = \sqrt{\frac{\ln(1000/0.05)}{2 \times 10^6}} = \sqrt{\frac{\ln(20000)}{2 \times 10^6}} \approx 0.0022 \quad (0.22\%).$$

Raising the threshold by 0.22% has a marginal impact on the objective so the certificate is close to free. The margin scales as $\epsilon \propto 1/\sqrt{|\mathcal{V}|}$, and feasibility evaluation is 2–13% of ACCESS runtime (Appendix I), so a tenfold larger $\mathcal{V}$ tightens the bound by $\sqrt{10} \approx 3.2$ at a cost that stays small. Table 2 compares the two bounds at the same empirical rate. Even granting SAA a multiplicity factor of one, which ignores the selection its own optimization performs, its bound sits 3.9% below the empirical rate against 0.22% for ours.

Table 2: Hoeffding lower bounds on true feasibility for an observed empirical rate of $\hat{p} = 0.99$, with $\delta = 0.05$.

Method	$\lvert\mathcal{V}\rvert$	Empirical $\hat{p}$	Hoeffding lower bound
SAA-favorable ($S = 1000$, $N = 1$)	1,000	0.99	0.951
ACCESS ($\lvert\mathcal{V}\rvert = 10^6$, $N = 1000$)	1,000,000	0.99	0.988

Note on guarantee scope. This bounds feasibility, not optimality. We do not claim the returned solution is close to the true chance-constrained optimum, only that its feasibility probability admits the stated lower bound.

D ALGORITHMIC DETAILS

This section gives the hyperparameters and the empirical behaviour of the search over (γ, β).

D.1 HYPERPARAMETER SETTINGS

Table 3 summarizes the hyperparameters used for ACCESS in all experiments. The total number of MILP solves per instance is $N_{\text{scen}} \times (N_{\text{init}} + N_{\text{iter}}) + 40 = 1000 \times 15 + 40 = 15{,}040$, of which the selection in line 10 of Algorithm 1 retains one per scenario, giving $N = 1000$ candidates for selection. Each solve is single-threaded with a 3 s limit, and scenarios are processed in parallel.

Table 3: ACCESS hyperparameter settings used in all experiments.

Parameter	Value	Description
Scenario Sampling		
N_{scen}	1000	Scenarios drawn from the problem's own distribution $\mathbb{P}$
Bayesian Optimization (per scenario)		
N_{init}	5	Initial Sobol samples
N_{iter}	10	BO iterations after initialization
Acquisition	qLogEI	Log expected improvement, feasibility-constrained
Search Space		
γ range	$[0,\ 2\hat{\gamma}_0]$	Tightening; $\hat{\gamma}_0$ calibrated per instance, capped at 20
β range	$[0,\ 1]$	Mixing weight, nominal to sampled scenario
Bracket points	40	Nominal solves calibrating $\hat{\gamma}_0$
Subproblem		
Time limit	3 s	Per MILP solve, single thread
MILP gap	10^{-4}	Gurobi default
Evaluation		
N (candidates)	1000	Number of candidate solutions for final selection
$\lvert\mathcal{V}'\rvert$	10^4	Scenarios scoring each point within a scenario's search
$\lvert\mathcal{V}\rvert$	10^6	Scenarios selecting among the N pooled candidates
$\lvert\mathcal{T}\rvert$	10^6	Held-out scenarios for evaluation (disjoint from $\mathcal{V}/\mathcal{V}'$)
Certified Feasibility (optional)		
δ	0.05	Failure probability for the Hoeffding bound

D.2 ESTIMATING THE PER-ROW SCALE

This section details how the per-row scale $\hat{\boldsymbol{\sigma}}$ is computed. Appendix E.1 reports an ablation on its efficacy when the uncertainty is heterogeneous across constraints. Throughout, $\mathcal{S}_\sigma = \{\boldsymbol{\xi}^{(1)}, \ldots, \boldsymbol{\xi}^{(n_\sigma)}\}$ is a set of n_σ scenarios drawn from $\mathbb{P}$ for this estimate alone, disjoint from the scenarios used to generate candidates and from $\mathcal{V}'$, $\mathcal{V}$ and $\mathcal{T}$.

Left-hand-side uncertainty. In the context of our experiments, we consider multiplicative left-hand-side uncertainty, and measure for each row how far its coefficients move relative to their nominal values,

$$r_{d,i}(\boldsymbol{\xi}) \;=\; \frac{\mathbf{A}(\boldsymbol{\xi})_{d,i} - \bar{\mathbf{A}}_{d,i}}{|\bar{\mathbf{A}}_{d,i}|}, \qquad \hat{\sigma}_d \;=\; \frac{1}{n}\sum_{i=1}^{n} \sqrt{\frac{1}{|\mathcal{S}_\sigma| - 1} \sum_{\boldsymbol{\xi} \in \mathcal{S}_\sigma} \Big(r_{d,i}(\boldsymbol{\xi}) - \bar{r}_{d,i}\Big)^2}, \tag{20}$$

where i indexes the n decision variables and d the rows of the joint constraint, $\mathbf{A}(\boldsymbol{\xi})_{d,i}$ is the coefficient of variable i in row d under scenario $\boldsymbol{\xi}$, $\bar{\mathbf{A}}_{d,i}$ is that same coefficient in the nominal scenario, and $\bar{r}_{d,i}$ is the mean of $r_{d,i}$ over $\mathcal{S}_\sigma$. Each row therefore carries the average relative volatility of its coefficients, measured in units of their nominal values.

Generalizing beyond left-hand-side uncertainty. The same computation applies to the right-hand side, with $r_{d,i}$ replaced by $(b(\boldsymbol{\xi})_d - \bar{b}_d)/|\bar{b}_d|$. A formulation carrying uncertainty on one side only

uses the corresponding scale directly. However, when both are uncertain, we need to combine the two deviations to construct a single scale per row, e.g., by summing them.

D.3 BO Convergence

In this section, we analyze the performance as a function of the number of iterations of BO. Within a scenario, the running best is the best objective among the points solved so far whose quick feasibility clears α. Figure 4 plots it after k BO iterations as a percentage over the best of the initial quasi-random samples, averaged over all instance/scenario sets. Generally, BO increases with the number of iterations, with some exceptions, e.g., SRCSP at low α, where the initialization was good enough to find high-quality solutions.

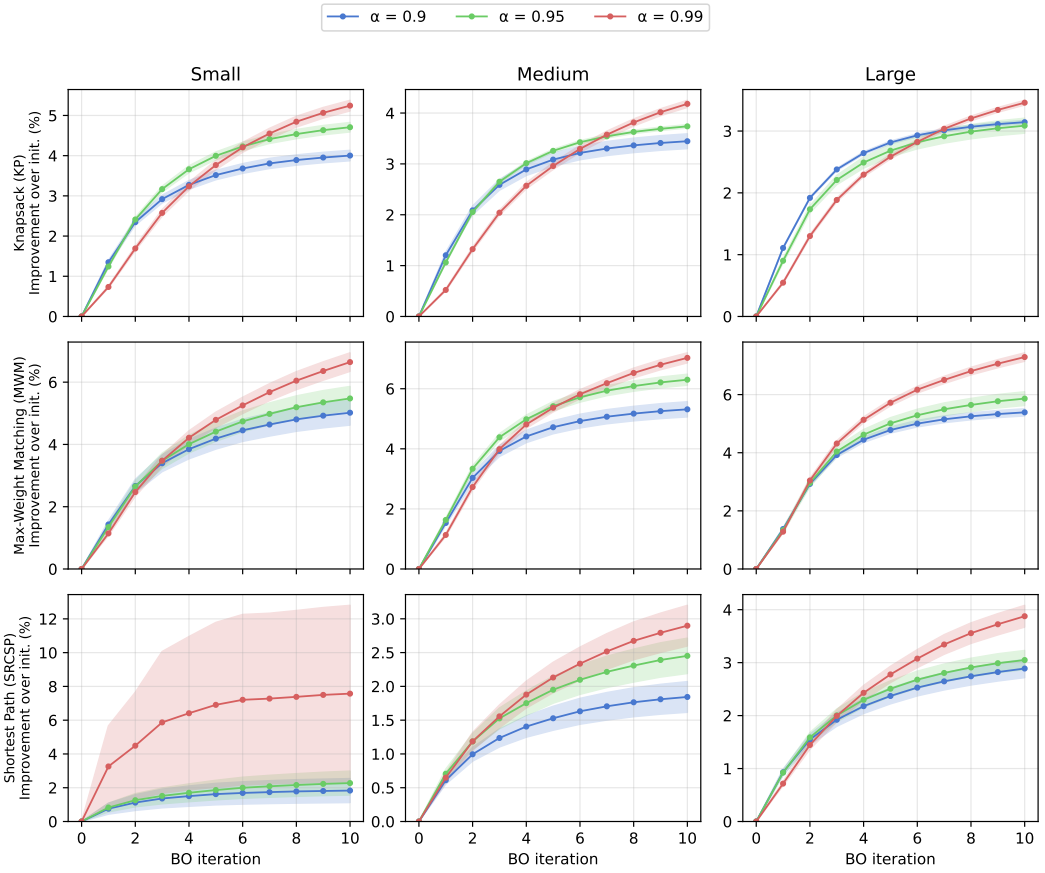

Figure 4: Improvements based on the number of BO iterations. Rows are benchmarks and columns are instance sizes. Iteration 0 is the end of the initial design, so the curve starts at 0% by construction. Shaded regions are the standard error across the 20 instances.

D.4 Scaling with Candidate Budget

Figure 5 shows how solution quality (measured by FO-gap) scales with the number of candidates N. For each instance, we subsample the first N candidates from the full ACCESS run and report the FO-gap relative to the best feasible objective at $N{=}1000$. The curves show rapid initial improvement followed by diminishing returns. SRCSP converges almost immediately, reaching the full-pool objective by $N{=}50$ in every cell. KP and MWM need more candidates but close to convergence on average by $N{=}100$, although individual instances still benefit from more iterations.

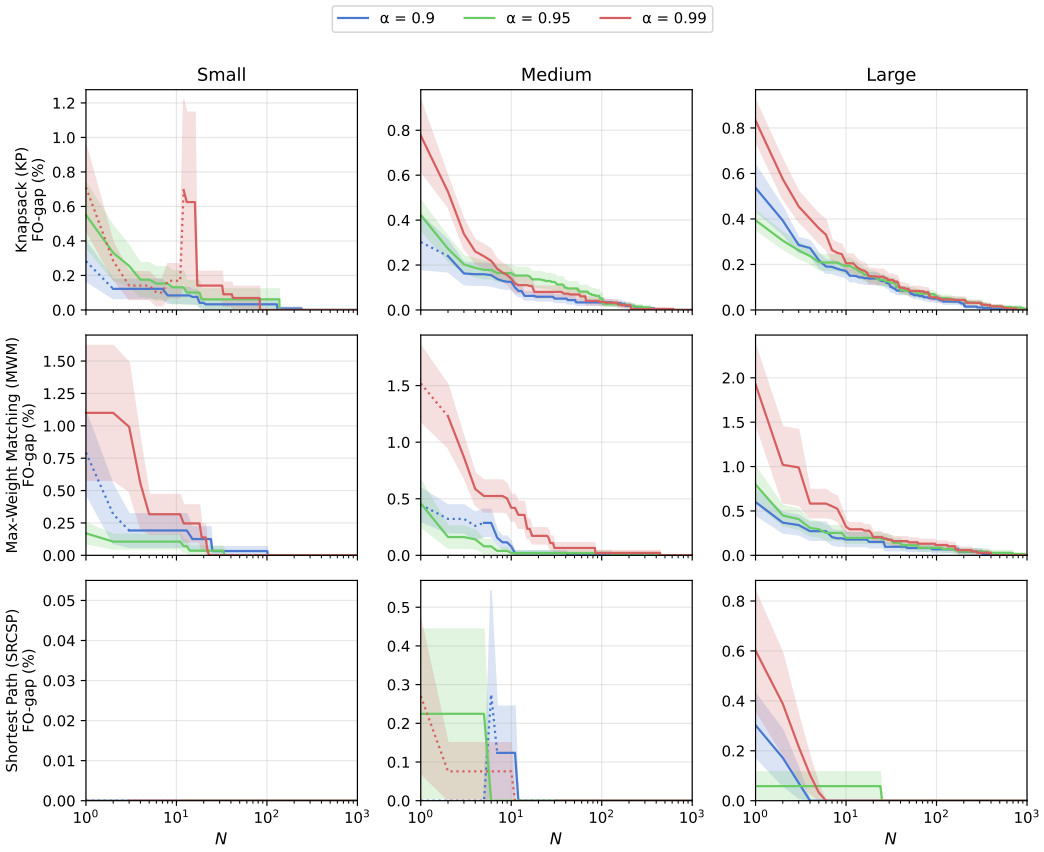

Figure 5: FO-gap vs. number of candidates N (log scale). Rows correspond to benchmarks (KP, MWM, SRCSP) and columns to instance sizes (Small, Medium, Large). FO-gap is computed relative to the best FO at $N{=}1000$. Higher α (red) generally requires more candidates to achieve low FO-gap. Shaded regions show standard error across 20 test instances.

D.5 Parameter Space Analysis

Figure 6 shows the selected parameters $(\gamma^\star, \beta^\star)$ for each test instance, where $(\gamma^\star, \beta^\star)$ are the parameters of the candidate chosen by the certified rule. The scatter plots reveal distinct patterns across benchmarks. For KP, the selected $\gamma^\star$ concentrates in $[0.3, 0.8]$ with $\beta^\star$ close to zero. For MWM, $\gamma^\star$ is small with $\beta^\star$ near zero at $\alpha \in \{.90, .95\}$ and rising at $\alpha{=}.99$. Lastly, for SRCSP both parameters spread more broadly, although this may be due to the instances being easier and having more viable good solutions.

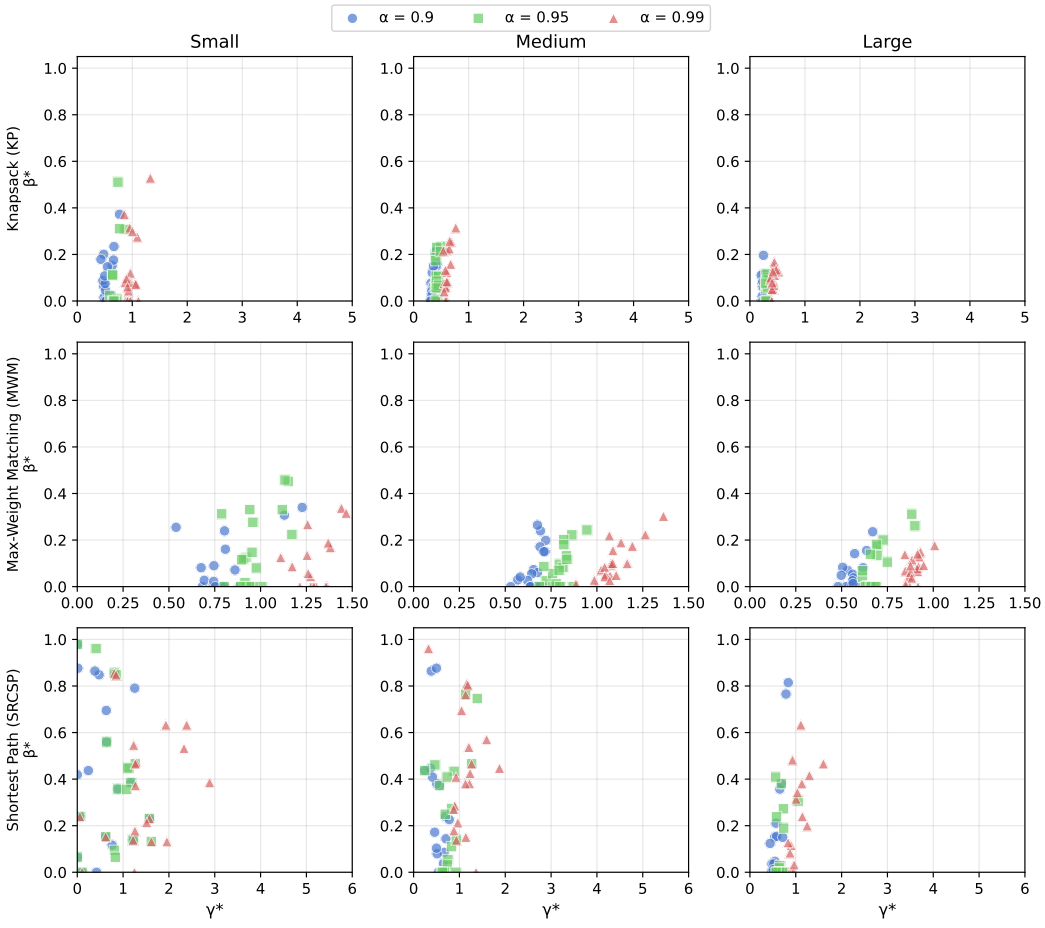

Figure 6: Selected parameters $(\gamma^\star, \beta^\star)$ per instance. Each point represents one test instance; colors denote $\alpha \in \{0.90, 0.95, 0.99\}$. Rows correspond to benchmarks and columns to instance sizes.

E NON-STANDARD UNCERTAINTY

The benchmarks of Appendix A use a single homogeneous σ and i.i.d. Gaussian ξ_i, which is what the classical safe approximations assume. ACCESS needs neither. It draws scenarios from the problem's own distribution and scores candidates by their empirical feasibility, so neither the search nor the selection rule refers to the distribution. We test this in two ways. Appendix E.1 varies the scale across constraints and keeps $\boldsymbol{\xi}$ Gaussian, and Appendix E.2 replaces the distribution itself with four non-Gaussian alternatives. The instances, seeds and multiplicative model are those of Appendix A throughout, and only the uncertainty changes.

E.1 HETEROGENEOUS UNCERTAINTY ABLATION

ACCESS normalizes constraint slack by a per-row σ_d, so rows with different noise levels are tightened differently. We compare against a variant that uses the single scalar $\bar{\sigma} = \frac{1}{d}\sum_{i=1}^{d}\sigma_i$ instead. We run the baselines on the same instances, so the cost of the scalar simplification can be read against what they pay rather than against ACCESS alone.

We evaluate on four KP variants at medium size (n=50, d=5), which differ only in $\boldsymbol{\sigma}$ and share the same mean within each pair:

- **hom_20**: $\boldsymbol{\sigma} = (0.20, 0.20, 0.20, 0.20, 0.20)$, i.e., the Medium KP of the main results for reference
- **hom_35**: $\boldsymbol{\sigma} = (0.35, 0.35, 0.35, 0.35, 0.35)$
- **het_20**: $\boldsymbol{\sigma} = (0.02, 0.11, 0.20, 0.29, 0.38)$
- **het_35**: $\boldsymbol{\sigma} = (0.02, 0.185, 0.35, 0.515, 0.68)$

The heterogeneous rows are evenly spaced about the same mean as their homogeneous counterpart, so a scalar $\bar{\sigma}$ is the correct mean in every setting and only its spread changes.

Table 4 reports the comparison. ACCESS performs similarly across the four settings as it consistently computes best-known feasible. The scalar-$\bar{\sigma}$ variant matches it on the homogeneous settings, where the two normalizations coincide, and degrades as the spread grows and as α tightens, since one $\bar{\sigma}$ misprices the rows that bind. At its worst it pays 10.23% against GB's 0.68% on the same instances. The per-row σ_d is therefore not a refinement, and justified to handle more general classes of uncertainty.

E.2 NON-GAUSSIAN UNCERTAINTY

The mixture of equation 9 makes no distributional assumption: it requires only that constraint data can be mixed, not that $\boldsymbol{\xi}$ be Gaussian, independent or light-tailed. To test this we repeat the knapsack benchmark under four alternative distributions. The instances, the seeds, and the multiplicative model $\mathbf{A}(\boldsymbol{\xi})_{d,i} = w_{d,i}\max(1+\sigma_d\xi_i,\,0)$ are identical to the Gaussian benchmark. Only the distribution of $\boldsymbol{\xi} \in \mathbb{R}^n$ changes, and each one below is zero-mean.

- **Student-t_3**, heavy tails with independent coordinates:

$$\xi_i = t_i/\sqrt{\nu/(\nu-2)}, \qquad t_i \overset{\text{iid}}{\sim} t_\nu, \quad \nu = 3.$$

- **Jump**, rare disruptions with a discrete right-skewed component:

$$\xi_i = z_i + J\,(B_i - p), \qquad z_i \overset{\text{iid}}{\sim} \mathcal{N}(0,1), \quad B_i \overset{\text{iid}}{\sim} \text{Bernoulli}(p), \quad p = 0.03,\ J = 6,$$

  with z and B independent. A coefficient jumps by roughly $\sigma_d J$ with probability p.
- **Clustered**, correlation aligned with the objective:

$$\xi_i = \sqrt{\rho}\,f_{c(i)} + \sqrt{1-\rho}\,z_i, \qquad f_1, f_2, z_i \overset{\text{iid}}{\sim} \mathcal{N}(0,1), \quad \rho = 0.85,$$

  where f_1 and f_2 are one factor per cluster and $c(i) \in \{1,2\}$ splits the items by value-to-mean-weight ratio, and cluster 1 holds the $n/2$ most attractive. The alignment is deliberate: the items a solution wants to take are the ones whose uncertainty moves together, so diversifying costs objective value.
- **Risky subset**, item scales that differ within a row:

$$\xi_i = \begin{cases} s\left(\sqrt{\rho}\,f + \sqrt{1-\rho}\,z_i\right), & i \in \mathcal{R}, \\ z_i, & \text{otherwise}, \end{cases} \qquad f, z_i \overset{\text{iid}}{\sim} \mathcal{N}(0,1), \quad s = 3,\ \rho = 0.8,$$

Table 4: Heterogeneous uncertainty on the knapsack benchmark (n=50, d=5). Each cell reports FO-gap / α-feas rate over 20 seeds, with feasibility measured on 10^6 held-out scenarios. The four settings share instances and seeds and differ only in the per-row σ_d; hom_20 is Medium KP from the main results.

	Method	α=.90	α=.95	α=.99
hom_20	SAA (S=1k)	0.21% / 40%	0.19% / 25%	— / 0%
	SAA (S=10k)	0.12% / 90%	0.04% / 75%	0.03% / 30%
	CVaR (S=1k)	4.16% / 100%	3.40% / 100%	1.37% / 85%
	CVaR (S=10k)	4.51% / 100%	4.02% / 100%	3.12% / 100%
	GB	3.00% / 100%	2.65% / 100%	2.14% / 100%
	ACCESS (scalar $\bar{\sigma}$)	0.00% / 95%	0.04% / 100%	0.00% / 100%
	ACCESS	0.00% / 95%	0.04% / 100%	0.00% / 100%
	ACCESS(δ)	0.06% / 100%	0.11% / 100%	0.39% / 100%
hom_35	SAA (S=1k)	0.37% / 15%	0.57% / 5%	— / 0%
	SAA (S=10k)	0.21% / 90%	0.28% / 70%	0.18% / 35%
	CVaR (S=1k)	6.65% / 100%	5.08% / 100%	1.72% / 60%
	CVaR (S=10k)	7.52% / 100%	6.58% / 100%	4.87% / 100%
	GB	5.04% / 100%	4.38% / 100%	3.40% / 100%
	ACCESS (scalar $\bar{\sigma}$)	0.06% / 100%	0.03% / 100%	0.00% / 90%
	ACCESS	0.04% / 100%	0.03% / 100%	0.00% / 90%
	ACCESS(δ)	0.15% / 100%	0.17% / 100%	0.65% / 100%
het_20	SAA (S=1k)	0.02% / 90%	0.03% / 75%	0.01% / 40%
	SAA (S=10k)	0.15% / 100%	0.07% / 90%	0.00% / 95%
	CVaR (S=1k)	0.86% / 100%	0.72% / 100%	0.46% / 100%
	CVaR (S=10k)	0.89% / 100%	0.76% / 100%	0.65% / 100%
	GB	0.58% / 100%	0.46% / 100%	0.36% / 100%
	ACCESS (scalar $\bar{\sigma}$)	0.10% / 100%	0.43% / 100%	2.93% / 90%
	ACCESS	0.00% / 100%	0.00% / 95%	0.00% / 100%
	ACCESS(δ)	0.02% / 100%	0.02% / 100%	0.07% / 100%
het_35	SAA (S=1k)	0.11% / 60%	0.05% / 60%	0.03% / 25%
	SAA (S=10k)	0.44% / 95%	0.24% / 95%	0.00% / 50%
	CVaR (S=1k)	1.31% / 100%	1.15% / 100%	0.75% / 100%
	CVaR (S=10k)	1.36% / 100%	1.32% / 100%	1.10% / 100%
	GB	0.78% / 100%	0.73% / 100%	0.68% / 100%
	ACCESS (scalar $\bar{\sigma}$)	2.00% / 100%	4.63% / 95%	10.23% / 100%
	ACCESS	0.00% / 100%	0.00% / 100%	0.02% / 95%
	ACCESS(δ)	0.01% / 100%	0.06% / 100%	0.17% / 100%

where $\mathcal{R}$ is a fixed random 30% of the items, drawn from the instance seed. The risky items have standard deviation s and share the single factor f; the rest keep the benchmark scale and stay independent.

Each distribution stresses a different assumption. Student-t_3 removes light-tail structure, jump introduces a skewed discrete component with no Gaussian analogue, clustering removes independence in the way that most penalises a per-row scalar tightening, and the risky subset gives the items within a row scales that differ by a factor of three. We omit the GB baseline: its per-row quantile presumes i.i.d. standard Gaussian ξ_i, and each distribution violates a different part of that. SAA and CVaR are distribution-free in construction and draw from the same scenario generator, so they transfer unchanged.

Table 5 reports every method on the four distributions. ACCESS(δ) is α-feasible on all 480 instances under every distribution, while retaining small gaps, and the uncorrected rule attains 0.00–0.32% at 80–100% feasibility. The sampled baselines degrade as expected with SAA not finding feasible solutions on large high α instances, SAA is feasible on 0–60% of instances at $|\mathcal{S}|$=10^3 and on 20–95% at 10^4, while CVaR stays feasible at large $|\mathcal{S}|$ but is worse in objective. In some cases, CVaR falls below α due to the sample estimate rather than the bound: the constraint is safe for the true tail mean, but we impose it on a sample average, and at the tightest α the Bonferroni split leaves too

Table 5: Non-Gaussian uncertainty on the knapsack benchmark. Each cell reports FO-gap / α-feas rate over 20 seeds, with feasibility measured on 10^6 held-out scenarios from the same distribution.

		n=50, d=5			n=100, d=10		
	Method	α=.90	α=.95	α=.99	α=.90	α=.95	α=.99
Student-t_3	SAA (S=1k)	0.13% / 40%	0.15% / 25%	— / 0%	0.00% / 10%	— / 0%	— / 0%
	SAA (S=10k)	0.44% / 80%	0.12% / 55%	0.03% / 40%	0.35% / 95%	0.06% / 70%	0.02% / 25%
	CVaR (S=1k)	5.24% / 100%	4.77% / 100%	2.84% / 50%	3.86% / 100%	3.09% / 100%	1.19% / 20%
	CVaR (S=10k)	6.31% / 100%	6.71% / 100%	7.80% / 100%	5.23% / 100%	5.43% / 100%	4.88% / 100%
	ACCESS	0.00% / 100%	0.01% / 100%	0.00% / 90%	0.00% / 100%	0.03% / 95%	0.02% / 85%
	ACCESS(δ)	0.05% / 100%	0.13% / 100%	0.71% / 100%	0.02% / 100%	0.07% / 100%	0.37% / 100%
Jump	SAA (S=1k)	0.32% / 15%	0.34% / 15%	— / 0%	0.02% / 15%	— / 0%	— / 0%
	SAA (S=10k)	0.04% / 55%	0.03% / 75%	0.00% / 20%	0.17% / 85%	0.04% / 55%	0.00% / 25%
	CVaR (S=1k)	7.44% / 100%	6.00% / 100%	1.81% / 50%	4.92% / 100%	3.49% / 100%	— / 0%
	CVaR (S=10k)	8.54% / 100%	7.85% / 100%	5.86% / 100%	6.50% / 100%	5.78% / 100%	3.77% / 100%
	ACCESS	0.06% / 95%	0.10% / 95%	0.05% / 95%	0.10% / 95%	0.13% / 95%	0.08% / 100%
	ACCESS(δ)	0.14% / 100%	0.24% / 100%	0.93% / 100%	0.13% / 100%	0.20% / 100%	0.59% / 100%
Clustered	SAA (S=1k)	0.48% / 50%	0.52% / 35%	0.71% / 30%	0.58% / 30%	0.61% / 25%	0.56% / 15%
	SAA (S=10k)	0.07% / 70%	0.07% / 80%	0.15% / 40%	0.09% / 70%	0.08% / 45%	0.06% / 55%
	CVaR (S=1k)	10.93% / 100%	9.26% / 100%	5.24% / 100%	12.53% / 100%	10.35% / 100%	5.62% / 95%
	CVaR (S=10k)	10.84% / 100%	9.36% / 100%	6.91% / 100%	12.64% / 100%	10.90% / 100%	8.22% / 100%
	ACCESS	0.14% / 100%	0.19% / 100%	0.07% / 90%	0.09% / 100%	0.10% / 95%	0.07% / 80%
	ACCESS(δ)	0.21% / 100%	0.31% / 100%	0.80% / 100%	0.17% / 100%	0.22% / 100%	1.03% / 100%
Risky subset	SAA (S=1k)	0.12% / 60%	0.06% / 40%	— / 0%	0.00% / 30%	0.00% / 20%	0.00% / 5%
	SAA (S=10k)	1.21% / 85%	0.43% / 90%	0.09% / 80%	1.44% / 95%	0.32% / 75%	0.00% / 35%
	CVaR (S=1k)	4.15% / 100%	3.40% / 100%	1.55% / 95%	3.12% / 100%	2.28% / 100%	0.69% / 75%
	CVaR (S=10k)	4.42% / 100%	3.88% / 100%	2.93% / 100%	3.47% / 100%	2.92% / 100%	2.01% / 100%
	ACCESS	0.05% / 100%	0.06% / 100%	0.20% / 100%	0.12% / 100%	0.20% / 100%	0.32% / 100%
	ACCESS(δ)	0.09% / 100%	0.08% / 100%	0.51% / 100%	0.15% / 100%	0.23% / 100%	0.61% / 100%

few scenarios in each row's tail to estimate it. The mixture gains most on these distributions as well (Appendix F).

F ABLATION: NOMINAL-ONLY TIGHTENING (β=0)

The scenario mixture is what separates ACCESS from a purely parametric tightening, so we isolate its contribution against the strongest nominal reference the formulation admits. The β=0 variant runs the same pipeline, i.e., the same γ bracket and the same selection rule, with the constraint data fixed at its nominal value. Nothing then varies across scenarios, so a grid over γ exhausts the search and we use 100 points. The ablation therefore measures the advantage of searching over (γ, β) per sampled scenario rather than over the tightening alone.

On the Gaussian benchmarks, Table 6 shows ACCESS at or below 0.04% FO-gap in every cell and ahead of or level with the β=0 variant throughout. Table 7 extends the comparison to the non-Gaussian distributions, where the advantage is larger. The nominal-only variant tightens rows by a single scalar, so it can identify how much rows vary but not how. In contrast, a sampled scenario is a vector, which captures how the uncertainty affects each variable. Under homogeneous Gaussian noise, the gains of including the scenario are marginal. Student-t_3 and the clustered distribution are scaled to unit variance, so that scalar is unchanged from the Gaussian case, yet the gain roughly doubles: heavy tails and correlation hurt the nominal tightening even though they do not change the scalar that sizes it. On the risky subset it reaches $+16.6\%$, where a fixed 30% of the items carry three times the noise of the rest, and a single scalar cannot distinguish the two groups. The median $\beta^\star$ is 0.36 on the risky subset and no instance there sets $\beta^\star = 0$, against a median of 0.06–0.07 on Gaussian knapsack and matching, where about a third do.

Table 6: Ablation: ACCESS vs. the nominal-only (β=0) variant, both under the uncertified rule. Rows report FO-gap (%) against BKO; Δ is the mean paired objective difference (positive = ACCESS ahead).

		Small			Medium			Large		
	Method	α=.90	α=.95	α=.99	α=.90	α=.95	α=.99	α=.90	α=.95	α=.99
KP	ACCESS	0.00	0.01	0.00	0.00	0.04	0.00	0.01	0.01	0.00
	β=0	0.09	0.18	0.11	0.19	0.28	0.26	0.34	0.33	0.37
	Δ	+0.09	+0.17	+0.11	+0.20	+0.24	+0.26	+0.33	+0.33	+0.37
MWM	ACCESS	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	β=0	0.19	0.17	0.43	0.35	0.21	0.73	0.26	0.39	0.66
	Δ	+0.20	+0.17	+0.44	+0.35	+0.20	+0.74	+0.26	+0.40	+0.63
SRCSP	ACCESS	0.00	0.00	0.00	0.03	0.00	0.00	0.04	0.00	0.00
	β=0	0.00	0.00	0.00	0.15	0.00	0.05	0.04	0.06	0.04
	Δ	+0.00	+0.00	+0.00	+0.13	+0.00	+0.05	-0.00	+0.06	+0.04

Table 7: Mean paired objective gain of ACCESS over the nominal-only (β=0) variant, both under the uncertified rule. Win, tie and loss count the instances on which ACCESS is ahead, equal and behind.

Benchmark	Instances	Mean gain	Win	Tie	Loss
KP	180	+0.23%	114	65	1
MWM	180	+0.38%	67	113	0
SRCSP	180	+0.03%	5	174	1
Student-t_3	120	+0.38%	104	14	2
Jump	120	+0.72%	117	3	0
Clustered	120	+0.81%	110	4	6
Risky subset	120	+16.60%	120	0	0

G BASELINE ANALYSIS

G.1 SAA

Table 8 reports FO-gap and α-feasibility rate for SAA at sample sizes $|\mathcal{S}| \in \{10, 100, 1000, 10000\}$. Small samples hold in-sample but fail on held-out data, while large ones (10^4) reach the time limit and return a conservative incumbent, more often feasible but at a larger gap.

Table 8: SAA results varying sample size $|\mathcal{S}|$. Each cell reports FO-gap / α-feas rate. Time limit: 3600 s.

		Small			Medium			Large		
	$\|\mathcal{S}\|$	α=.90	α=.95	α=.99	α=.90	α=.95	α=.99	α=.90	α=.95	α=.99
KP	10	—/0%	—/0%	—/0%	—/0%	—/0%	—/0%	—/0%	—/0%	—/0%
	100	0.14%/20%	0.00%/10%	—/0%	—/0%	—/0%	—/0%	—/0%	—/0%	—/0%
	1000	0.08%/75%	0.11%/50%	0.00%/45%	0.21%/40%	0.19%/25%	—/0%	0.11%/25%	0.29%/5%	—/0%
	10000	0.10%/85%	0.10%/70%	0.04%/80%	0.12%/90%	0.04%/75%	0.03%/30%	0.19%/95%	0.14%/70%	0.06%/30%
MWM	10	—/0%	—/0%	—/0%	—/0%	—/0%	—/0%	—/0%	—/0%	—/0%
	100	0.41%/45%	0.00%/35%	0.00%/15%	0.13%/15%	0.00%/5%	—/0%	0.55%/10%	—/0%	—/0%
	1000	0.09%/75%	0.17%/75%	0.13%/55%	0.80%/85%	0.62%/55%	0.11%/55%	1.38%/65%	0.84%/80%	0.04%/40%
	10000	3.49%/95%	3.08%/95%	0.52%/95%	4.26%/95%	4.28%/100%	1.70%/95%	5.23%/95%	4.57%/100%	1.84%/95%
SRCSP	10	0.00%/55%	2.09%/50%	0.00%/20%	2.31%/15%	1.68%/20%	—/0%	—/0%	0.00%/15%	—/0%
	100	0.00%/90%	0.13%/90%	0.00%/80%	0.12%/85%	1.17%/80%	0.39%/45%	0.11%/55%	0.18%/40%	0.00%/15%
	1000	1.84%/95%	0.00%/100%	0.00%/95%	0.00%/90%	0.98%/95%	0.79%/70%	0.07%/85%	0.28%/75%	0.41%/55%
	10000	1.74%/100%	0.00%/95%	0.00%/100%	0.52%/100%	0.09%/100%	0.00%/85%	3.58%/95%	2.09%/100%	0.69%/95%

G.2 CVAR

Table 9 reports the same for CVaR, which adds only continuous auxiliary variables and so reaches the hour far less often than SAA. Small samples leave one or two scenarios in each row's tail, so the estimate is not safe and α=0.99 fails even at $|\mathcal{S}|$=10^3 scenarios. Moreover, the Bonferroni allocation leads to overly conservative solutions, and the degradation relative to SAA is significant.

Table 9: CVaR results varying sample size $|\mathcal{S}|$. Each cell reports FO-gap / α-feas rate. Time limit: 3600 s.

		Small			Medium			Large		
	$\|\mathcal{S}\|$	α=.90	α=.95	α=.99	α=.90	α=.95	α=.99	α=.90	α=.95	α=.99
KP	10	—/0%	—/0%	—/0%	0.00%/5%	—/0%	—/0%	—/0%	—/0%	—/0%
	100	4.13%/100%	2.59%/80%	0.00%/5%	2.47%/90%	1.52%/30%	—/0%	1.33%/60%	—/0%	—/0%
	1000	5.82%/100%	4.88%/100%	3.33%/100%	4.16%/100%	3.40%/100%	1.37%/85%	3.25%/100%	2.45%/100%	0.89%/35%
	10000	6.00%/100%	5.26%/100%	4.39%/100%	4.51%/100%	4.02%/100%	3.12%/100%	3.81%/100%	3.34%/100%	2.38%/100%
MWM	10	0.00%/10%	—/0%	—/0%	—/0%	—/0%	—/0%	—/0%	—/0%	—/0%
	100	6.59%/100%	4.11%/95%	0.15%/25%	3.57%/100%	2.42%/80%	—/0%	2.60%/100%	1.58%/30%	—/0%
	1000	9.20%/100%	6.76%/100%	5.15%/100%	5.75%/100%	5.18%/100%	3.39%/95%	4.49%/100%	3.85%/100%	1.94%/100%
	10000	9.49%/100%	6.43%/100%	6.57%/100%	6.75%/100%	6.42%/100%	4.77%/100%	6.61%/100%	6.27%/100%	4.68%/100%
SRCSP	10	4.72%/80%	2.09%/50%	0.00%/20%	1.66%/60%	1.68%/20%	—/0%	0.42%/40%	0.00%/15%	—/0%
	100	10.97%/100%	7.40%/100%	1.50%/90%	7.62%/100%	5.91%/100%	1.74%/75%	6.00%/100%	3.77%/100%	0.00%/35%
	1000	10.38%/100%	7.16%/100%	3.10%/100%	8.56%/100%	6.84%/100%	6.95%/100%	8.62%/100%	7.44%/100%	4.62%/100%
	10000	11.31%/100%	7.86%/100%	7.92%/100%	8.17%/100%	7.59%/100%	6.73%/100%	8.89%/100%	7.93%/100%	6.47%/100%

H PER-INSTANCE DISTRIBUTIONAL RESULTS

This section reports the distribution of outcomes across the 20 instances for each benchmark, instance size, and α value. We compare ACCESS against the baseline methods: SAA ($|\mathcal{S}|$=1000), CVaR ($|\mathcal{S}|$=1000), and the GB approximation.

H.1 FEASIBILITY RATE VS. OPTIMALITY GAP

Figures 7–9 plot feasibility rate against optimality gap, with columns varying α and rows varying the problem size, and the dashed line marking the α threshold. Every solution is shown, feasible or not, and the gap is measured against the best α-feasible objective found by any method, the reference of Table 1, so an infeasible solution that beats it appears at a negative gap. The three benchmarks show the same pattern: ACCESS sits near 0% gap just above the threshold, SAA falls below it, and CVaR and GB clear it at a substantially larger gap.

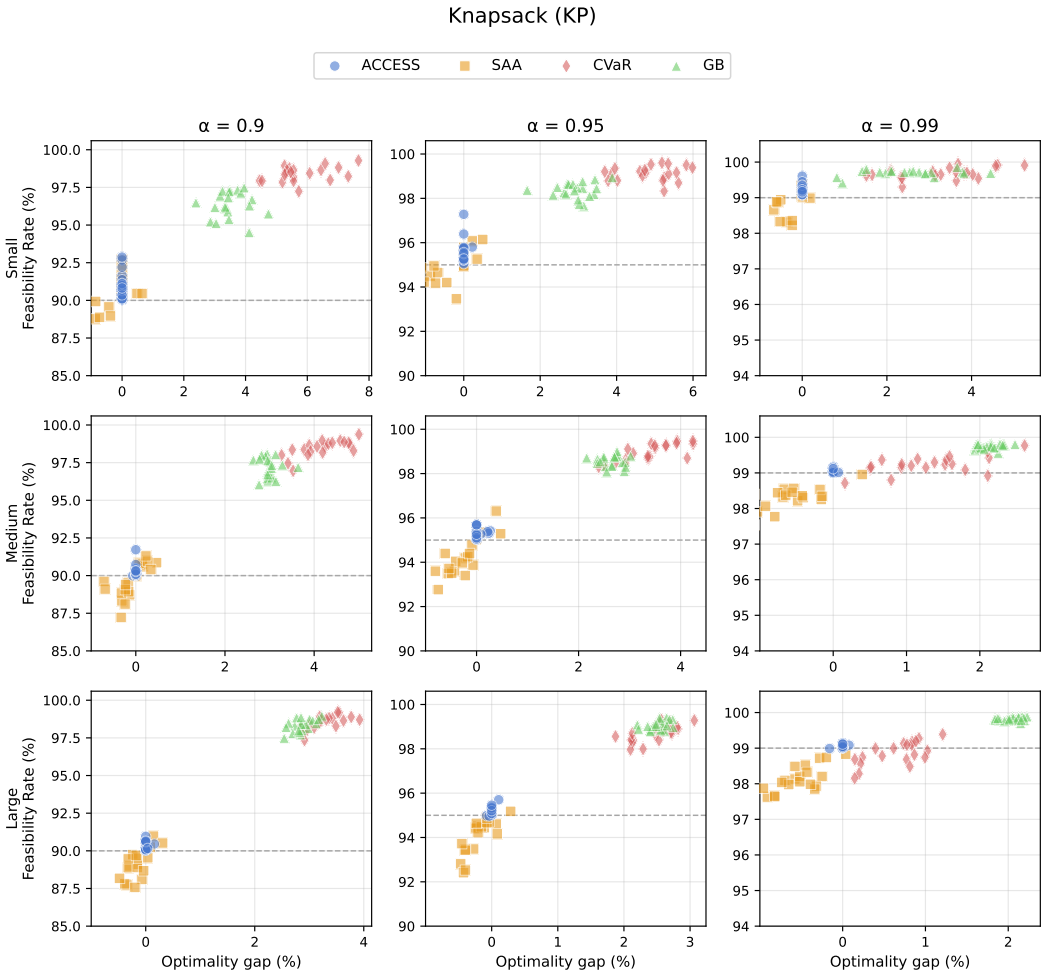

Figure 7: Per-instance feasibility rate versus optimality gap for the Knapsack Problem (KP). Each point is one of the 20 instances, feasible or not.

H.2 GENERATION TIME VS. FO-GAP

Figures 10–12 compare time against FO-gap, and include only α-feasible solutions, since runtime is not comparable for a solution that does not meet the constraint. Again the three benchmarks agree: ACCESS reaches near-zero FO-gap in about two minutes on KP and SRCSP and about ten on the

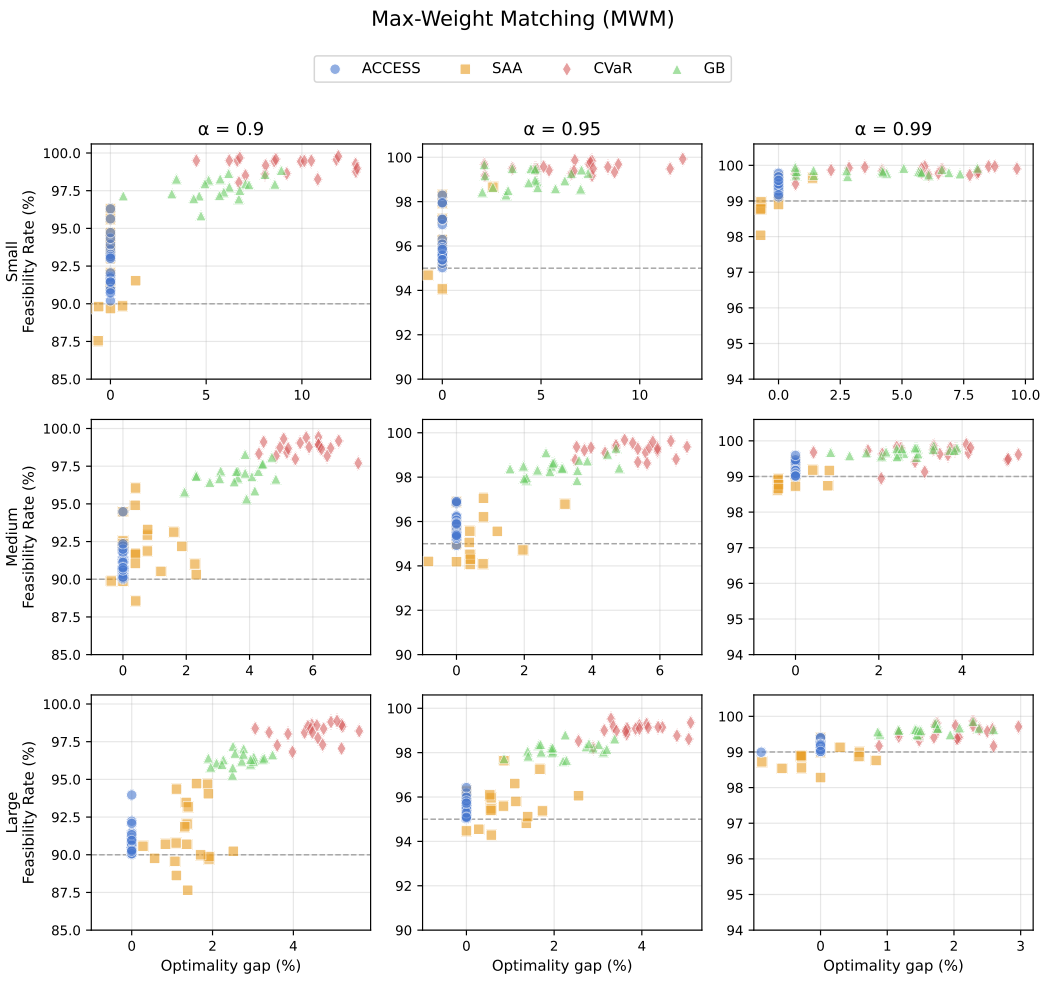

Figure 8: Per-instance feasibility rate versus optimality gap for Maximum Weight Matching (MWM).

largest MWM instances, SAA reaches the hour, and GB and CVaR are faster but pay a much larger gap.

H.3 Feasibility Distributions: ACCESS vs Random Sampling

Figures 13–15 compare the per-candidate feasibility (evaluated on 10^6 scenarios) of the ACCESS pool with that of $1{,}000$ random solutions per instance, on every instance size and α, pooled across the 20 test instances. Consistently, we can observe that ACCESS concentrates solutions near the boundary, while random sampling often produces completely feasible or infeasible solutions.

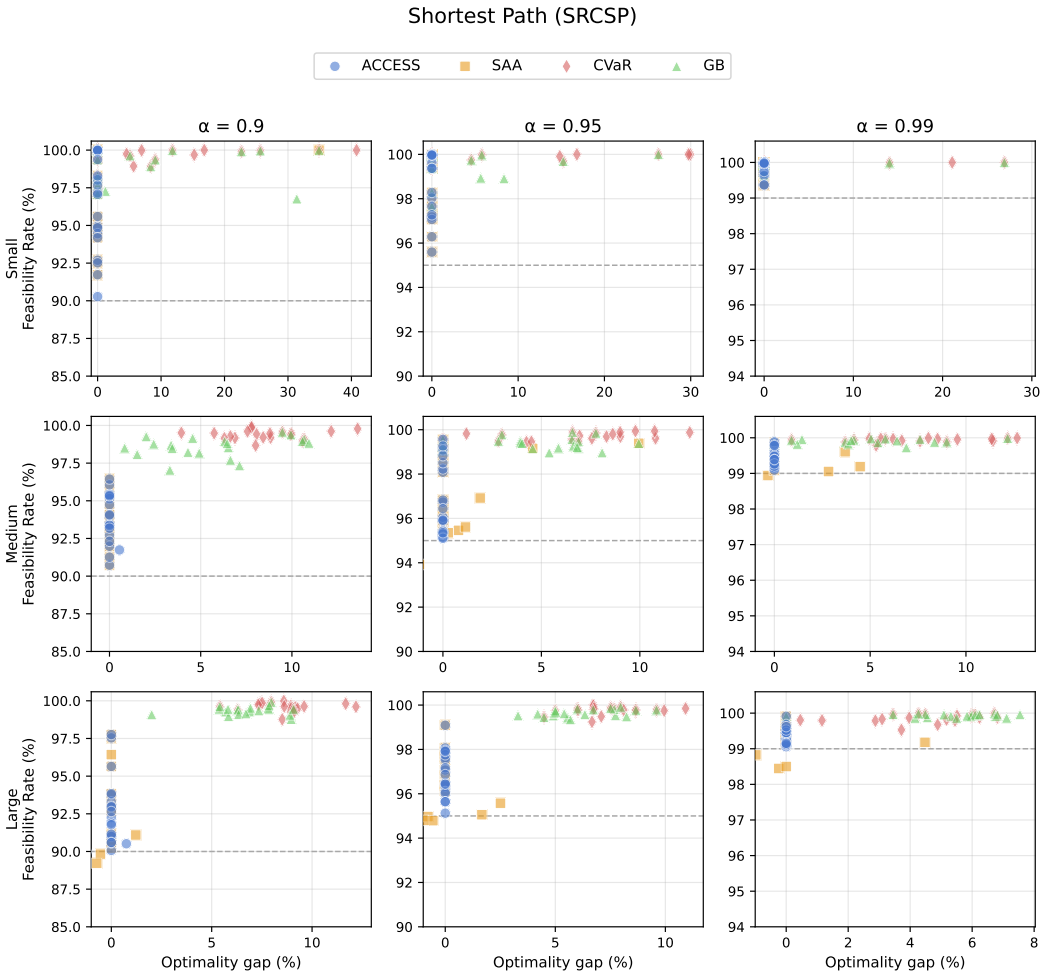

Figure 9: Per-instance feasibility rate versus optimality gap for Stochastic Resource-Constrained Shortest Path (SRCSP).

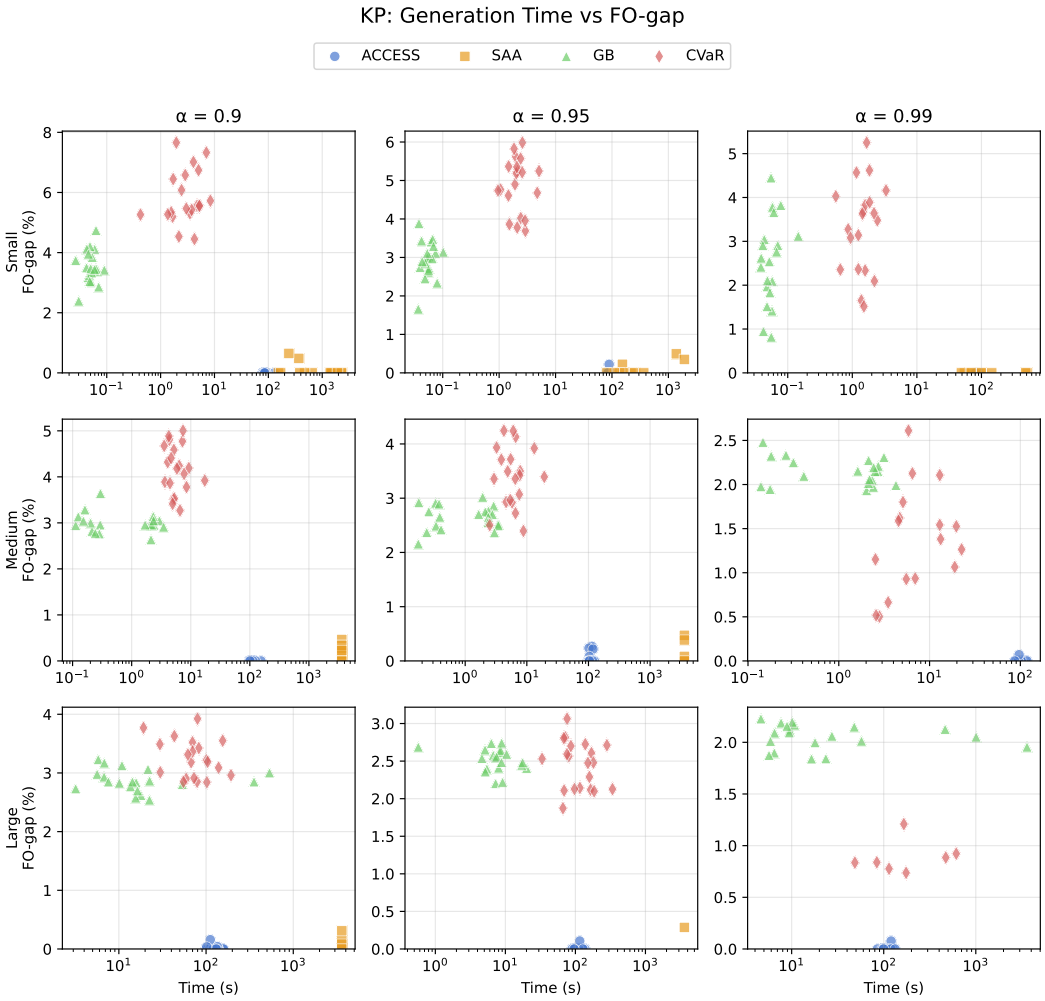

Figure 10: Generation time versus FO-gap for KP, over the α-feasible solutions only.

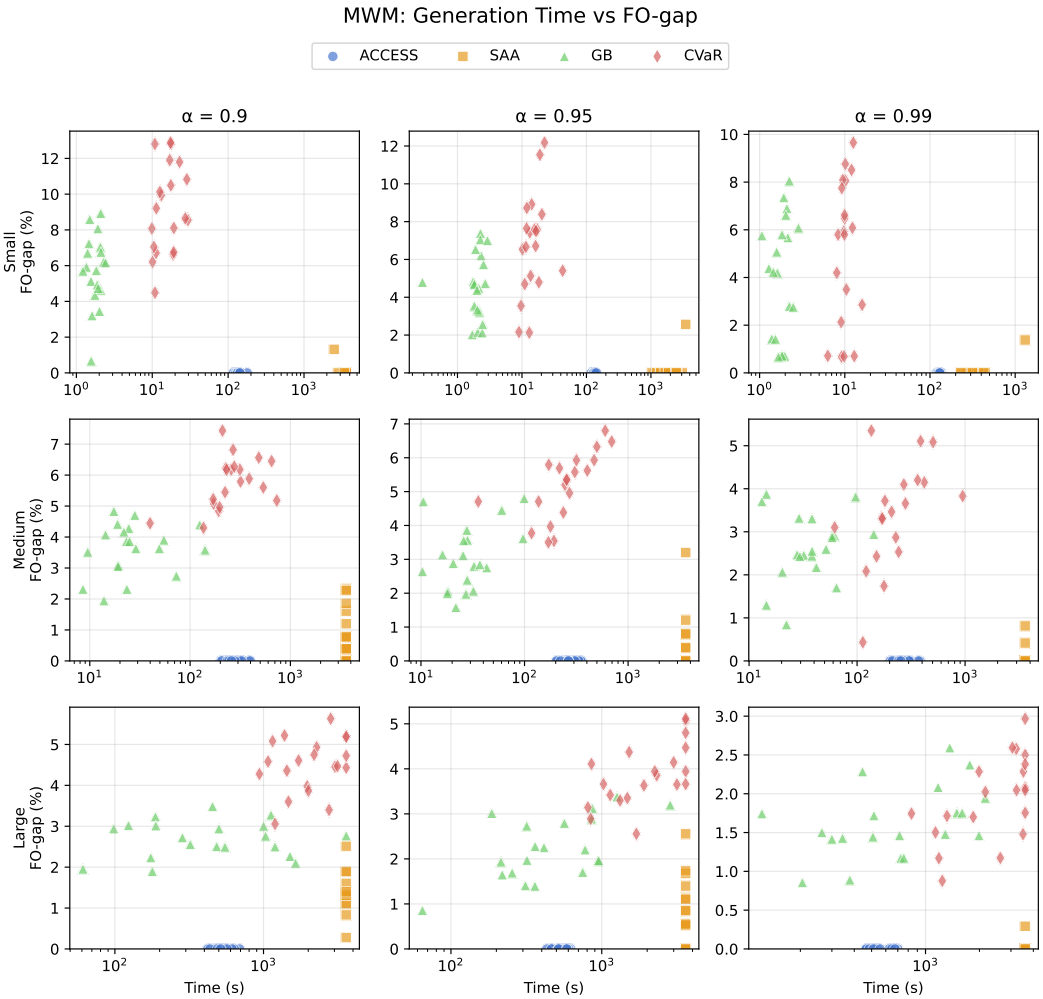

Figure 11: Generation time versus FO-gap for MWM, over the α-feasible solutions only.

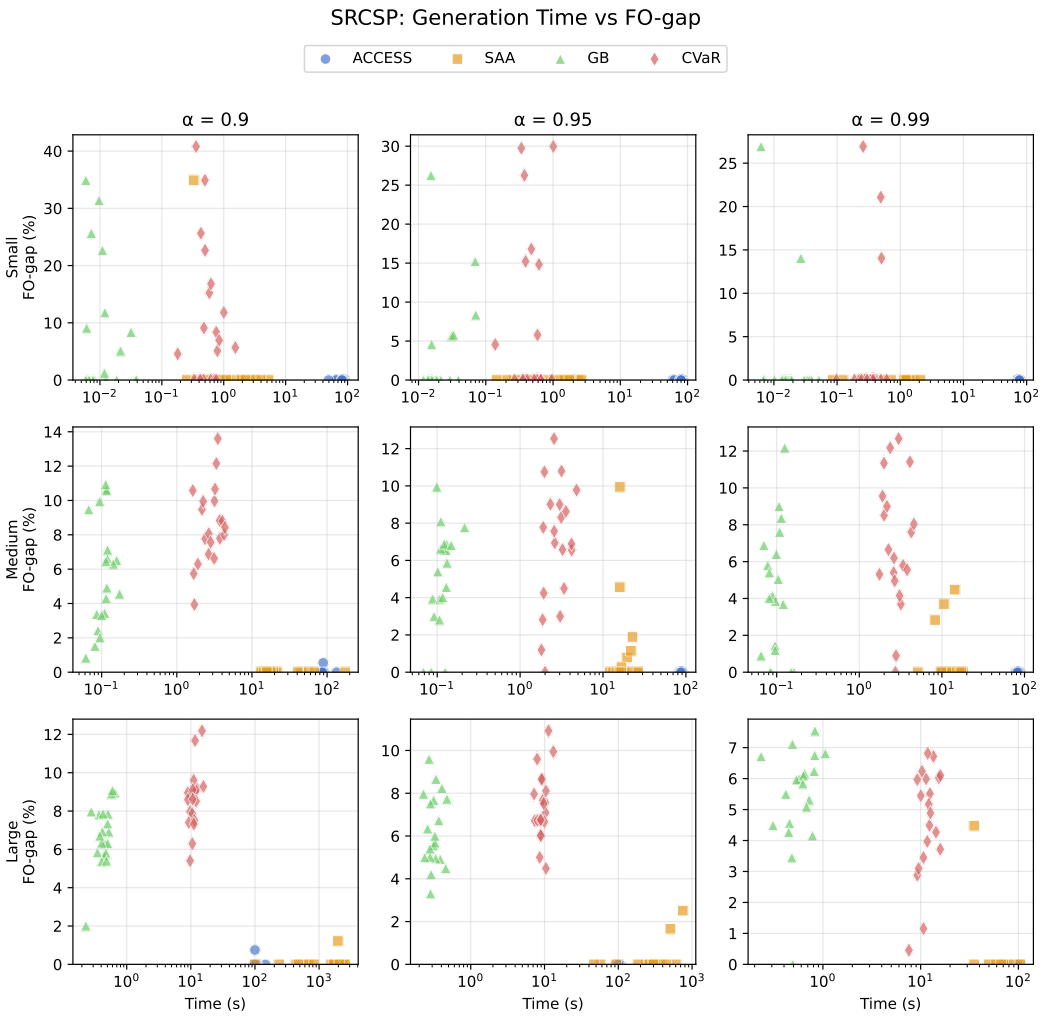

Figure 12: Generation time versus FO-gap for SRCSP, over the α-feasible solutions only.

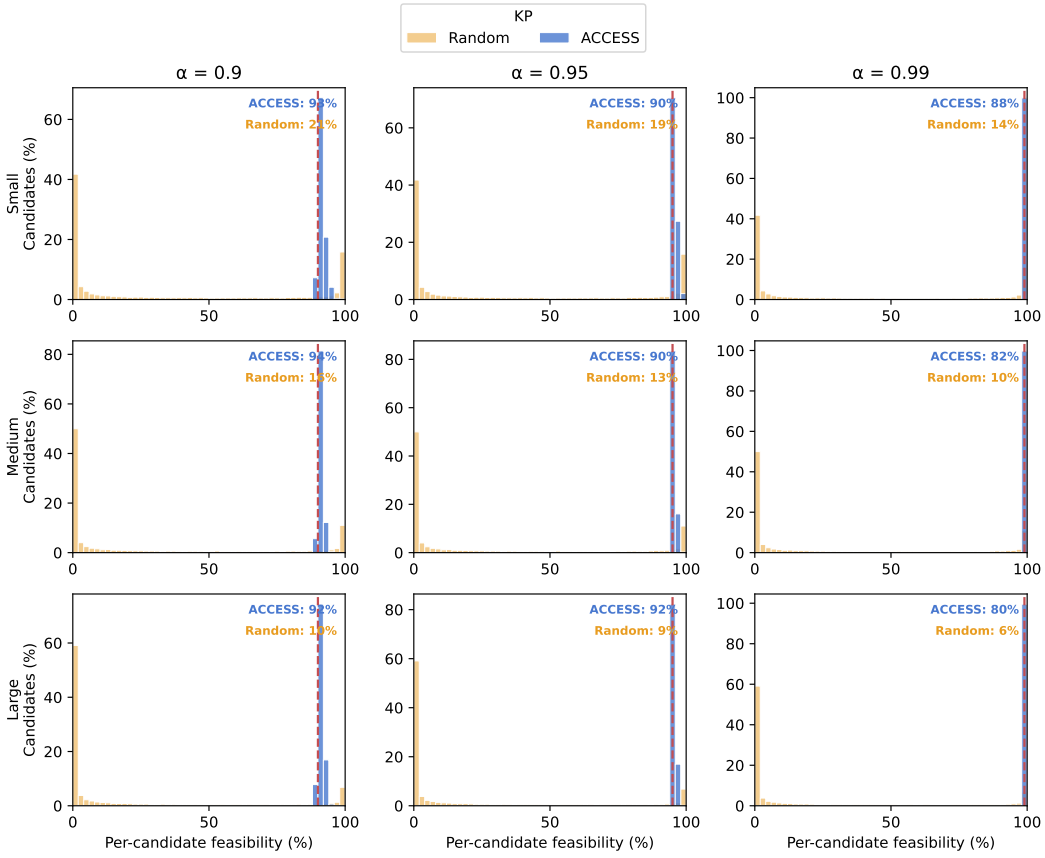

Figure 13: Feasibility distributions for Knapsack (KP). Rows are instance sizes and columns are α.

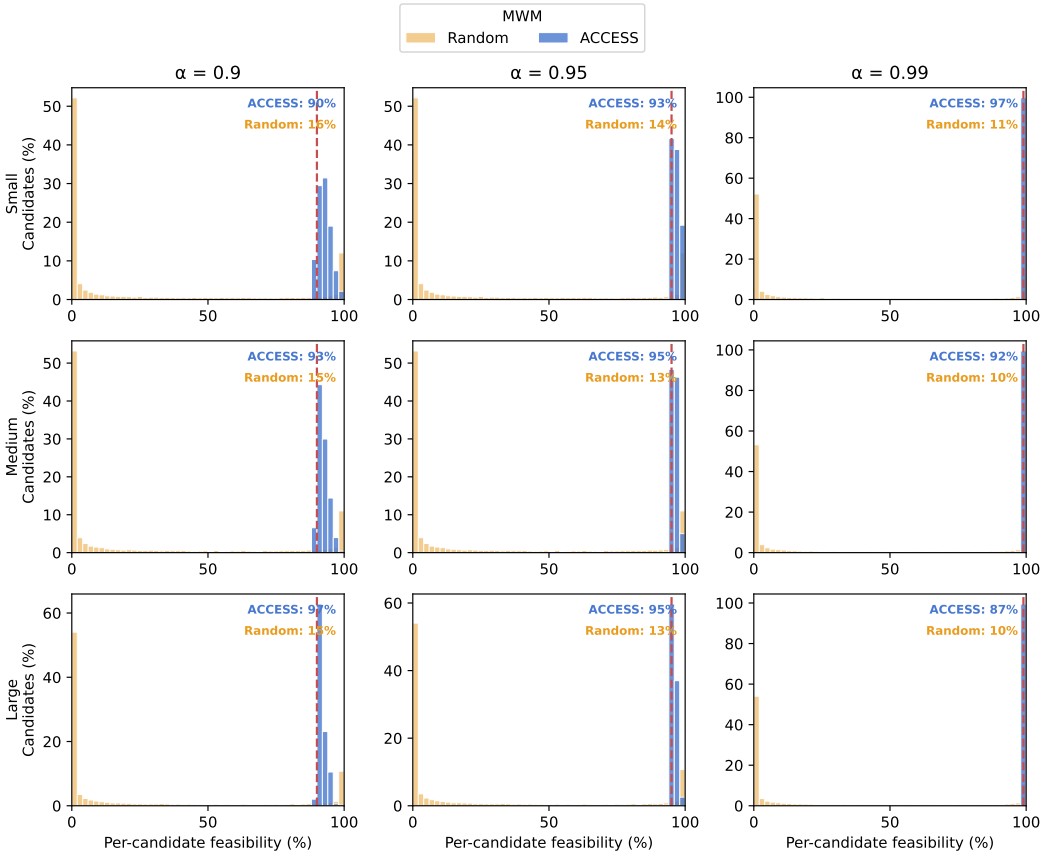

Figure 14: Feasibility distributions for Maximum Weight Matching (MWM).

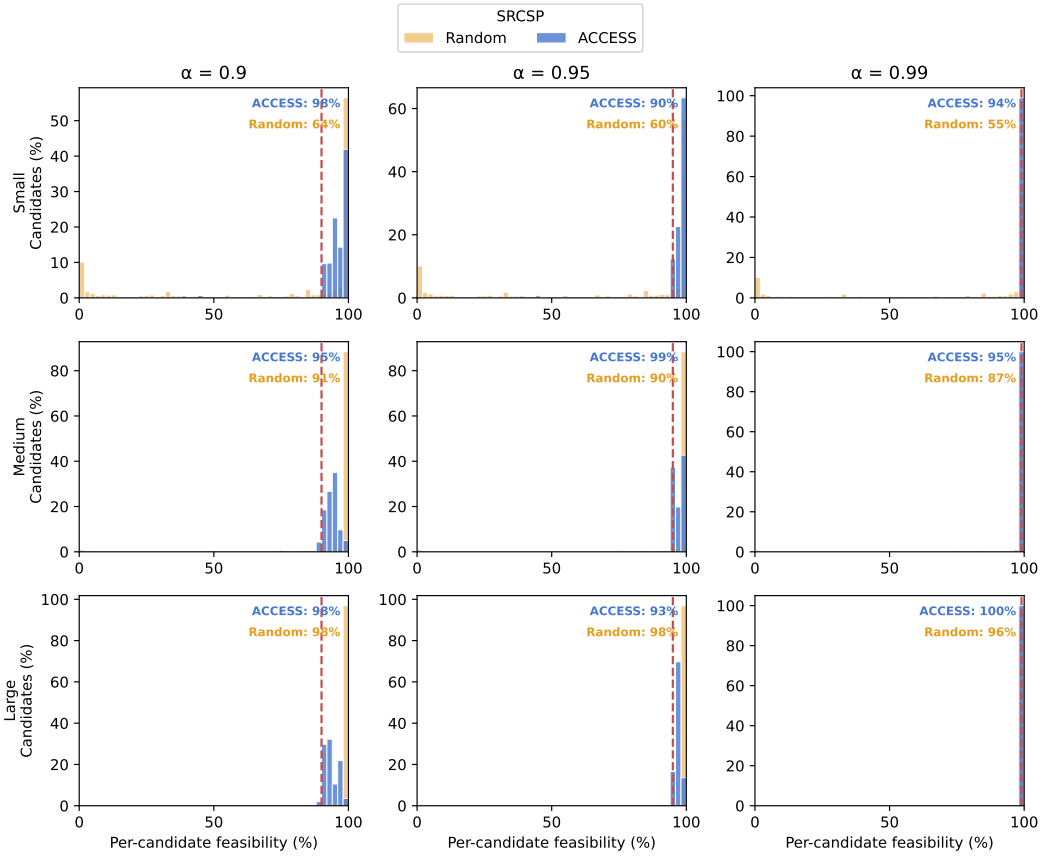

Figure 15: Feasibility distributions for Stochastic Resource-Constrained Shortest Path (SRCSP).

I Detailed Runtime Results

Table 10 provides detailed generation time statistics (mean across 20 test instances) for each benchmark, size, and α value. For ACCESS, times are shown as generation/feasibility to separate the BO search from the final constraint verification. Feasibility checking is 2–13% of the total, so a larger evaluation set, and with it a tighter certificate, costs little.

Table 10: Generation time in seconds (mean across 20 instances). ACCESS times shown as generation/feasibility.

		Small			Medium			Large		
	Method	α=.90	α=.95	α=.99	α=.90	α=.95	α=.99	α=.90	α=.95	α=.99
KP	ACCESS	87.3/2.7	85.7/2.8	81.6/2.8	111.1/2.7	109.8/2.8	103.2/2.8	122.9/16.1	109.6/15.9	106.0/15.8
	SAA	1082	440	199	3601	3601	3561	3600	3600	3600
	GB	0.1	0.1	0.1	1.3	1.6	1.7	58	8.6	267
	CVaR	3.5	2.3	1.6	6.3	6.6	9.1	81	130	256
MWM	ACCESS	131.3/6.7	130.7/6.4	127.0/6.7	281.6/15.3	276.5/14.8	271.3/14.4	526.7/22.9	543.2/23.6	557.3/22.4
	SAA	3397	2082	350	3600	3600	3600	3600	3600	3600
	GB	1.8	2.1	1.9	37	34	45	735	642	906
	CVaR	17	16	10	301	289	268	2249	2245	2622
SRCSP	ACCESS	81.4/2.5	79.3/2.5	75.6/2.6	89.7/3.5	87.7/3.2	85.2/3.3	103.0/9.0	99.9/8.9	96.7/9.1
	SAA	1.5	1.0	0.8	32	18	12	1293	294	66
	GB	0.0	0.0	0.0	0.1	0.1	0.1	0.5	0.3	0.6
	CVaR	0.6	0.5	0.3	2.9	2.9	2.9	11	9.4	12