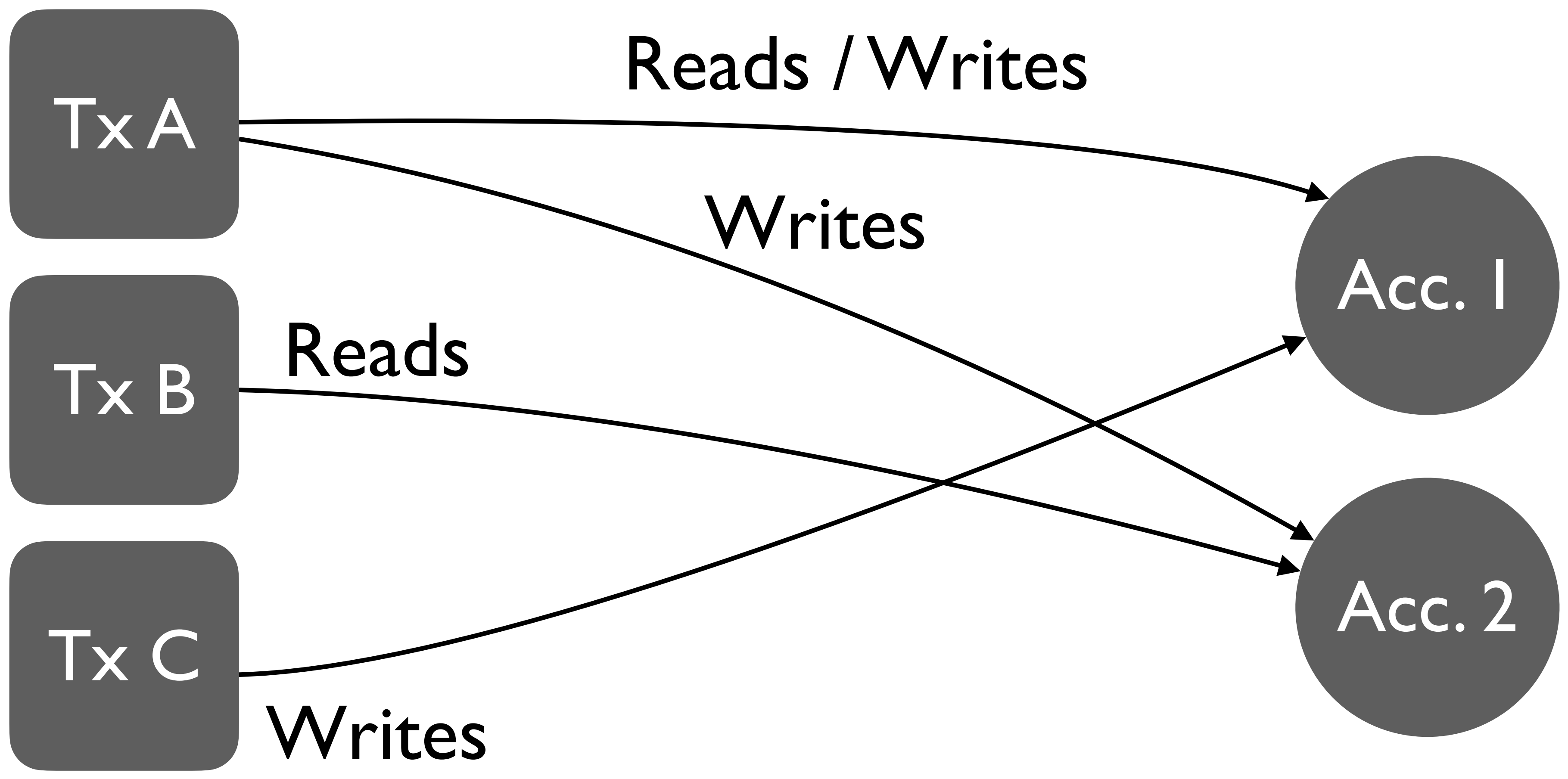
# A Lagrangian Approach to Conflict-aware Transaction Packing

Francesco Iannelli @ DeFi & Crypto Workshop

# Block building

- Block builders decide which transactions get selected for inclusion.
- Extracted fees and throughput are directly linked to:
  - validator incentives.
  - economic security.
- Block building is a security primitive:
  - revenue & throughput $\Longleftrightarrow$ security.
- As execution becomes parallel, selection quality matters more.

# Why conflicts matter




- Conflicts cause failures, wasted blockspace and hinder parallel execution $\implies$ cannot be ignored.
- Outright prohibition would leave money on the table.
- Greedy or overly conservative selection ignores this tradeoff.

# Existing approaches and their limits

- Greedy:
  - packs the block sorting transactions by fee over cost.
  - ignores conflicts.
- Greedy conflict-aware:
  - packs the block sorting transactions by fee over cost but only packs non-conflicting ones.
  - hard exclusions.

# Key idea: soft conflicts

- Conflicts are priced, not forbidden.
  - profit = revenue − conflict penalty.
- Kernelization.
  - Let $y_i, y_j \in \mathbb{R}^d$ be feature vectors representing transaction $i$ and $j$ with fees $q_i$ and $q_j$.
  - Let $\phi(\,\cdot\,,\cdot\,) \to [0,1]$ be a PSD kernel and let $\phi(y_i, y_j) = \Phi_{ij}$ be the pairwise conflict likelihood.
  - We define the penalty incurred when including both $i$ and $j$ as $Q_{ij} = \Phi_{ij} \min\{q_i, q_j\}$.

# Kernel instantiations

- Several PSD kernels may be employed:
  - linear, polynomial, Gaussian, etc…
- Weighted variants:
  - $W = \begin{bmatrix} W_{rr} & W_{rw} \\ W_{wr} & W_{ww} \end{bmatrix} \succeq 0.$
  - $$\phi(y_i, y_j) = \frac{y_i^T W y_j}{\sqrt{y_i^T W y_i}\sqrt{y_j^T W y_j}}, \quad \phi(y_i, y_j) = \exp\left(-\frac{1}{2\sigma}(y_i - y_j)^T W (y_i - y_j)\right).$$
  - weights can be learned from historical failure data.

# Modeling as a quadratic knapsack

- Compact formulation:

$$\begin{array}{ll} \text{maximize} & q^T x - \frac{\gamma}{2} x^T Q x \\ \text{subject to} & c^T x \le M, \quad x \in \{0,1\}^n. \end{array}$$

  - $q \in \mathbb{R}^n_+$ fees.
  - $c \in \mathbb{R}^n_+$ costs.
  - $M > 0$ block capacity.
  - $Q \in \mathbb{S}^n_+$ conflict penalties.
  - $\gamma \in [0,1]$ risk-revenue tradeoff parameter.

# Quadratic term proxies expected loss

- Let $F_{ij}$ let the binary RV indicating that transaction $i$ fails due to a conflict with $j$.
- Let $F_i = \bigvee_j F_{ij}$ be the binary RV indicating whether transaction $i$ fails.
- Let $F = \{F_i\}_{i=1}^n$ and the total loss $L(x; F) = \sum_{i=1}^n q_i x_i F_i$.
- Applying the union bound and linearity of expectation we get:
  - $\mathbb{E}_F[L(x; F)] \leq \dfrac{1}{2} \sum_{1 \leq i,j \leq n} \Phi_{ij} \min\{q_i, q_j\} x_i x_j = \frac{1}{2} x^T Q x.$
  - we assume only the lower fee transaction fails in a conflict.

# Continuous relaxation

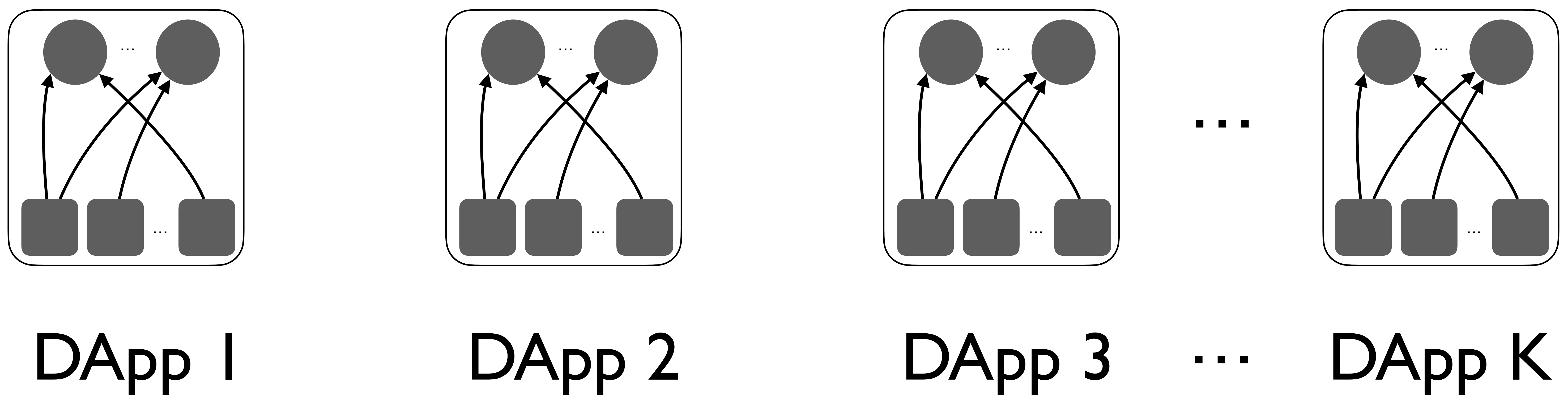
- Quadratic knapsack is NP-hard:
  - finding an exact solution is computationally difficult.
- If we relax the integrality constraint we obtain a tractable formulation:
  - Substitute $x \in \{0,1\}^n$ with $x \in [0,1]^n$,

$$\begin{aligned} &\text{maximize} && q^T x - \frac{\gamma}{2} x^T Q x \\ &\text{subject to} && c^T x \leq M, \quad x \in [0,1]^n . \end{aligned}$$

  - concave objective and convex feasible region $\Longrightarrow$ polynomial time solution.

# Conflict graph and decomposition

- Conflicts induce a graph structure.
- Each transaction interacts with a limited number of accounts.
- Access patterns form connected components.



# Lagrangian relaxation

- Dualize the capacity constraint.
  - $c^T x \leq M \rightarrow \lambda(c^T x - M), \quad \lambda \geq 0.$
- The original problem decomposes into $K$ independent sub-problems, one for each connected component.
- The dual function is defined as:

$$g(\lambda) = \sum_{k=1}^{K} \max_{x_{\mathcal{C}_k} \in [0,1]^{|\mathcal{C}_k|}} [(q_{\mathcal{C}_k} - \lambda c_{\mathcal{C}_k})^T x_{\mathcal{C}_k} - \frac{\gamma}{2} x_{\mathcal{C}_k}^T Q_{\mathcal{C}_k} x_{\mathcal{C}_k}] + \lambda M.$$

# Dual variable as price

- The dual variable $\lambda$ has a clear economic interpretation.
  - $\lambda =$ shadow price of compute.
  - $q_i - \lambda c_i \leq 0 \implies$ transaction $i$ is not included.
- Transactions compete on fee to cost adjusted for conflict penalties.

# Dual root finding

- Solving the dual reduces to the following root finding problem
  - $g'(\lambda) = M - c^T x^*(\lambda) = 0.$
- The dual derivative $g'(\lambda)$ is increasing and the root can be efficiently bracketed in $O(\log(1/\varepsilon))$ iterations.
- Cool, but what about integrality?

# Rounding: from fractional to integer

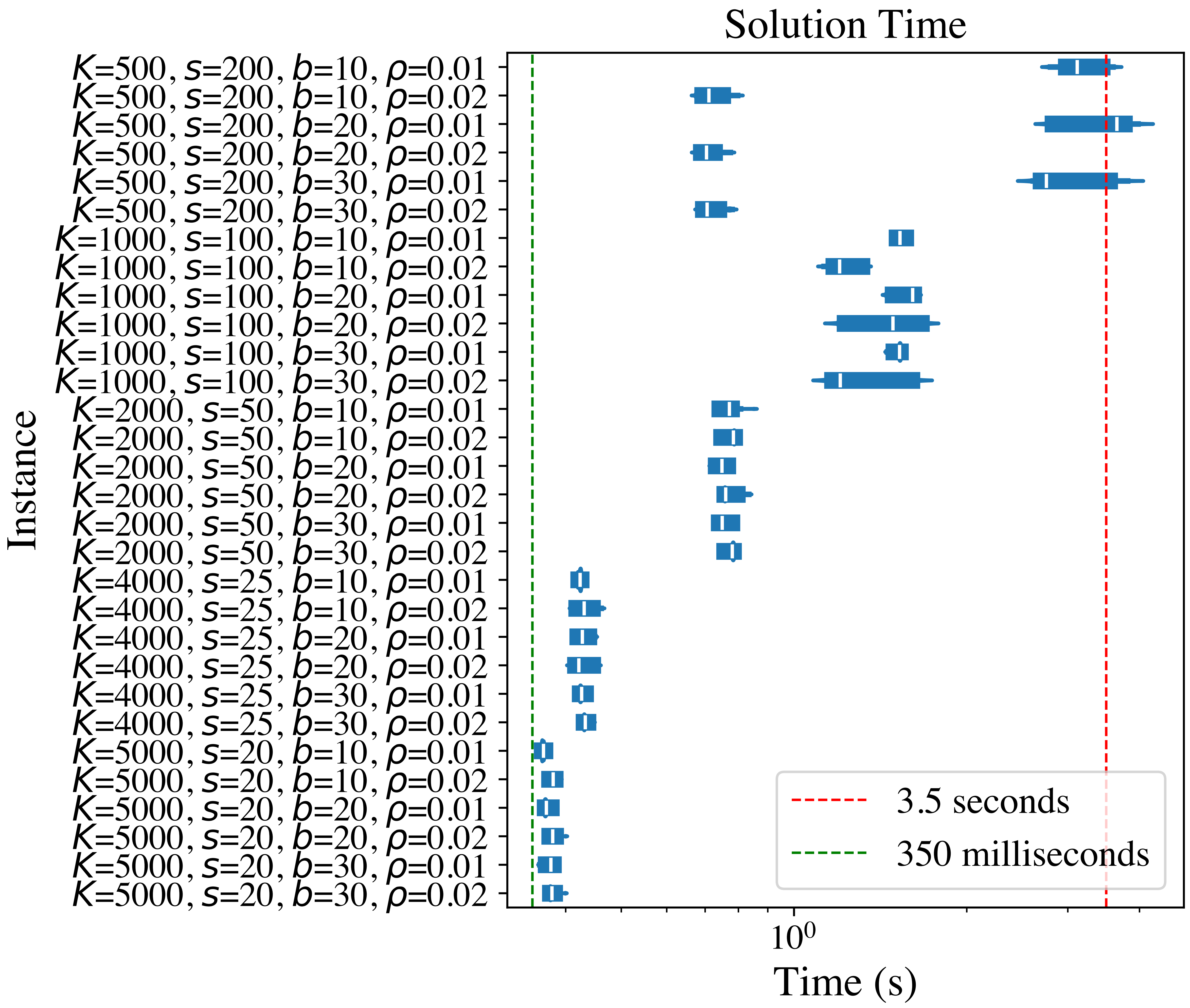
- Steps: $x^{\text{frac}} \to x^{\text{int}} \to x^{\text{feas}}$. Cost is $O(n \log n)$.

1. Bernoulli rounding
   - $x^{\text{frac}} \to x^{\text{int}}$.
   - $x_i^{\text{int}} \sim \text{Bernoulli}(x_i^{\text{frac}}), \quad i = 1, \ldots, n$.
2. Greedy pruning for feasibility ($c^T x \leq M$).
   - $x^{\text{int}} \to x^{\text{feas}}$.
   - Removes selected transactions with smallest fee-to-cost ratios until capacity is met.

# Guarantees

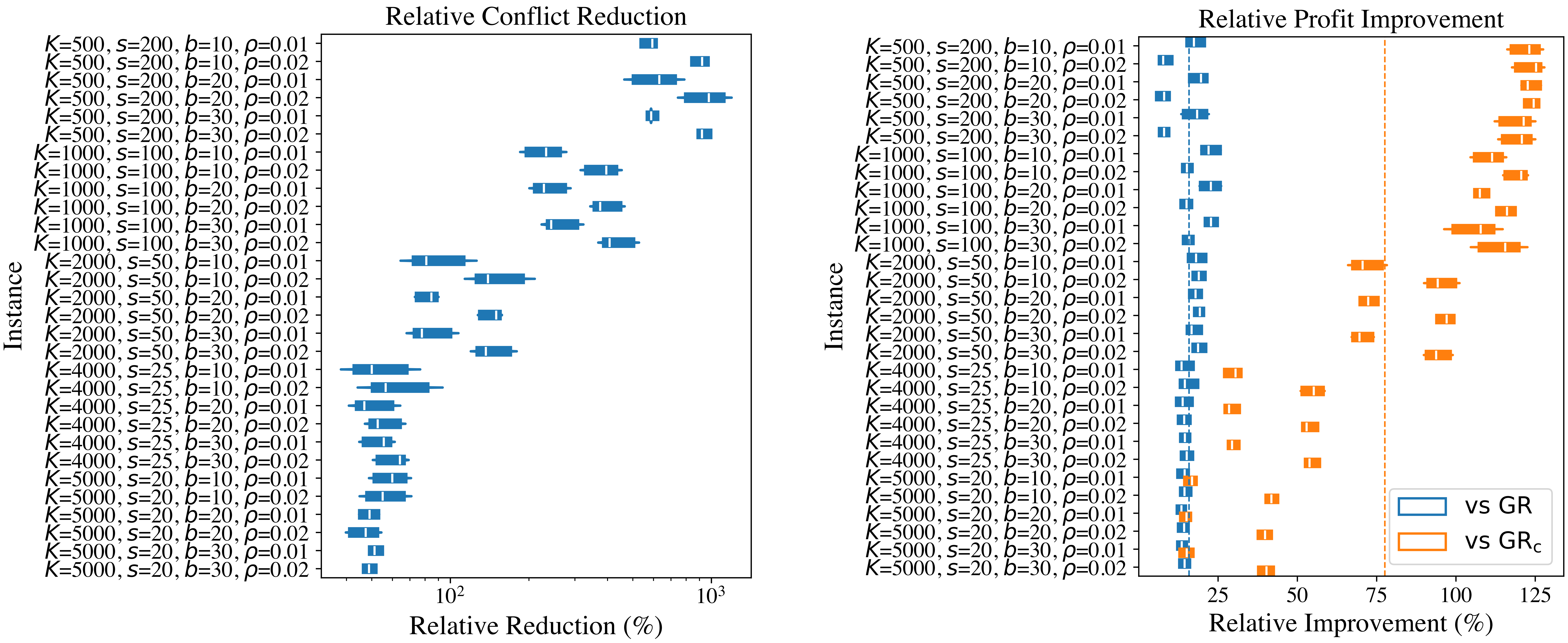
- High probability error bound. For any $\delta, \eta \in (0,1)$:
  - $$\Pr\left[f(x^{\text{feas}}) \geq f(x^{\text{frac}}) - \sqrt{\frac{U}{2}\ln\frac{1}{\delta}} - R\sqrt{\frac{V}{2}\ln\frac{1}{\eta}}\right] \geq 1 - \delta - \eta\,.$$
- The loss depends on:
  - Conflict intensity $U$.
  - Cost variability $V$.
  - The largest fee-to-cost ratio $R$.
- Rounded solutions retain at least 90% of the fractional optimum in tested instances.
- Repeating the rounding procedure in parallel boosts odds exponentially.

# Scalability




- The total number of TXs is approximately $K \times s$.
  - $K$ controls the number of connected components.
  - $s$ controls their sizes.
- $b$ controls the conflict intensity.
- $\rho$ controls capacity.
- 100k+ TXs are packed in seconds on consumer grade hardware.

# Quality vs. baselines



# Takeaways

- Soft conflicts beat naive methods:
  - naive greedy.
  - greedy with hard exclusions.
- Our method is principled:
  - scales by exploiting the problem's structure.
  - leverages modern multicore hardware.
- Practical for real block builders.

Link to preprint