Optimization by Parallel Quasi-Quantum Annealing with Gradient-Based Sampling

Yuma Ichikawa [1], Yamato Arai [1, 2]

[1]Fujitsu Limited [2]The University of Tokyo

Introduction

Sampling-based Solvers (e.g., Simulated Annealing)
Combinatorial optimization (CO) is interpreted as sampling from the corresponding distribution.
Limitation: Local and sequential updates limit scalability and parallelism.

Learning-based Solvers
The methods learn heuristics without the need for manual design and leveraging GPUs.

- **Supervised Learning (SL)**: Models are trained using labeled optimal solutions.
  Limitation Requires access high-quality labeled data (optimal solutions).
- **Reinforcement Learning (RL)**: Learns policies by optimizing reward signals in CO tasks.
  Limitation: Often suffers from training instability and difficulties in exploration.
- **Unsupervised Learning (UL)**: Optimizes model parameters to directly minimize CO objectives.
  Issue: Sensitive to model architecture and computationally expensive to train.

Recent Advances: Revisiting Sampling-based Solvers (iSCO [2])
iSCO integrates discrete Langevin dynamics with simulated annealing.

- iSCO achieves performance comparable to learning-based methods without training.
- Enables efficient GPU-parallel updates, overcoming the limitation of sampling-based methods.

Parallel Quasi-Quantum Annealing (PQQA): A Further Advancement in Sampling-based Solvers
PQQA combines gradient-based transitions, quasi-quantum annealing, and parallel exploration facilitated by inter-process communication.

- PQQA outperforms both learning-based solvers and iSCO, with performance improvements becoming increasingly significant as problem size grows.

Background

Combinatorial Optimization: The goal is to find an optimal discrete solution, formulated as follows:

$$\min_{\boldsymbol{x}\in\mathcal{X}(C)} f(\boldsymbol{x};C),\quad \mathcal{X}(C)=\left\{\boldsymbol{x}\in\{0,1\}^N \;\middle|\; \begin{array}{l} g_i(\boldsymbol{x};C)\le 0\ \forall i\in[I], \\ h_j(\boldsymbol{x};C)=0\ \forall j\in[J] \end{array}\right\}. \tag{1}$$

$C\in\mathcal{C}$: a problem-specific instance parameter (e.g., a graph $G=(V,E)$); f is the cost function to be minimized; $\mathcal{X}(C)$: the feasible set; and g_i, h_j: inequality and equality constraints, respectively.

Penalty Method: In sampling-based solvers, Eq. (1) is reformulated as an unconstrained problem:

$$\min_{\boldsymbol{x}\in\{0,1\}^N} l(\boldsymbol{x};C,\boldsymbol{\lambda}),\quad l(\boldsymbol{x};C,\boldsymbol{\lambda})\triangleq f(\boldsymbol{x};C)+\sum_{i=1}^{I+J}\lambda_i v_i(\boldsymbol{x};C),\quad \boldsymbol{\lambda}=(\lambda_i)_{i=1}^{I+J}\in\mathbb{R}^{I+J}. \tag{2}$$

$v_i(\cdot)$: a penalty that increases when the constraints are violated; $\boldsymbol{\lambda}$ denotes the penalty strengths.

$$\forall i\in[I],\quad v_i(\boldsymbol{x};C)=\max(0,g_i(\boldsymbol{x};C)),\ \forall j\in[J],\quad v_j(\boldsymbol{x};C)=(h_j(\boldsymbol{x};C))^2.$$

Continuous Relaxation: PQQA employs a continuous relaxation strategy defined as follows:

$$\min_{\boldsymbol{p}\in[0,1]^N} \hat{l}(\boldsymbol{p};C,\boldsymbol{\lambda}),\quad \hat{l}(\boldsymbol{p};C,\boldsymbol{\lambda})\triangleq \hat{f}(\boldsymbol{p};C)+\sum_{i=1}^{I+J}\lambda_i\hat{v}_i(\boldsymbol{p};C),$$

This approach has the following limitations:

- Relaxation Gap: There may be a significant discrepancy between the optimal solution of the original CO problem and that of its relaxed version.
- Ambiguity in Rounding: The relaxation often yields half-integral values $1/2$, which undermines the robustness of the existing rounding methods.

Quasi-Quantum Annealing (QQA)

Entropy Term: We introduce the following entropy term:

$$\hat{r}(\sigma(\boldsymbol{w});C,\boldsymbol{\lambda},\gamma)=\hat{l}(\sigma(\boldsymbol{w});C,\boldsymbol{\lambda})+\gamma s(\sigma(\boldsymbol{w})),\quad \boldsymbol{w}\in\mathbb{R}^N,\quad \sigma:\mathbb{R}^N\to[0,1]^N,\quad \gamma\in\mathbb{R}. \tag{3}$$

The entropy term $s(\sigma(\boldsymbol{w}))$ is a convex function that attains its minimum of 0 when $\sigma(\boldsymbol{w})$ and its maximum when $\sigma(\boldsymbol{w})=\mathbf{1}_N/2$. γ controls this trade-off. Specifically, α-entropy [1] is employed:

$$s(\sigma(\boldsymbol{w}))=\sum_{i=1}^{N}\left\{1-(2\sigma(w_i)-1)^{\alpha}\right\},\quad \alpha\in\{2n\mid n\in\mathbb{N}\}. \tag{4}$$

Relaxed Boltzmann Distribution: We define the relaxed Boltzmann distribution as follows:

$$\hat{P}(\sigma(\boldsymbol{w});\gamma,T)=\frac{1}{Z(\gamma,T)}e^{-\frac{1}{T}\hat{r}(\sigma(\boldsymbol{w});C,\boldsymbol{\lambda},\gamma)},\quad Z(\gamma,T)=\int_N d\boldsymbol{w}e^{-\frac{1}{T}\hat{r}(\sigma(\boldsymbol{w});C,\boldsymbol{\lambda},\gamma)}. \tag{5}$$

- $\gamma<\mathbf{0}$: The relaxed variables tend to favor half-integral values. **The convex entropy term helps to smooth the non-convexity of the relaxed objective function.**
- $\gamma>0$: The relaxed variables tend to favor discrete values. The relaxed distribution approaches the original discrete distribution.

Theorem

The relaxed Boltzmann distribution $\lim_{\gamma\to+\infty}\lim_{T\to0}\hat{P}(\sigma(\boldsymbol{w});\gamma,T)$ converges to a uniform distribution over the optimal solutions of Eq. (2), i.e., $\boldsymbol{x}^*\in\operatorname{argmin}_{\boldsymbol{x}} l(\boldsymbol{x};C,\boldsymbol{\lambda})$. Additionally, $\lim_{\gamma\to-\infty}\lim_{T\to0}\hat{P}(\sigma(\boldsymbol{w});\gamma,T)$ converges to a single-peaked distribution $\prod_{i=1}^{N}\delta(\sigma(w_i)-1/2)$.

Gradient-based Update: The following gradient-based update is applicable for any value of γ:

$$\boldsymbol{w}^{t+1}=\boldsymbol{w}^t-\eta\nabla_{\boldsymbol{w}}r(\sigma(\boldsymbol{w}^t);C,\boldsymbol{\lambda})+\sqrt{2\eta T}\boldsymbol{\xi},\quad \boldsymbol{\xi}\sim\mathcal{N}(\mathbf{0}_N,I_N) \tag{6}$$

In contrast to local updates used in conventional sampling-based solvers, **this method simultaneously updates multiple variables in a single gradient step**.

Annealing Strategy: The annealing strategy described below incrementally increases γ from a negative to a positive value during the update of $\boldsymbol{w}$.

- Initial Condition ($\gamma\le0$): encourages broad exploration by smoothing the non-convexity of $\hat{l}(\sigma(\boldsymbol{w});C,\boldsymbol{\lambda})$, favoring ***quasi-quantum state*** $\sigma(\boldsymbol{w})=\mathbf{1}_N/2$.
- γ is gradually increased to a positive value until the entropy approaches zero to round the relaxed variables by suppressing continuous suboptimal oscillating between 1 and 0.

This annealing resembles *quantum annealing*, where the system transitions from a superposition state of 0 and 1 to the classical state of the target optimization problem.

Communication in Parallel Runs

QQA enables efficient batch-parallel computation across multiple initializations and instances using GPUs. Therefore, the following communication is introduced across all S parallel runs, defined as $\sigma(W)\triangleq\{\sigma(\boldsymbol{w}^{(s)})\}_{s=1}^{S}$:

$$\mathcal{P}(\sigma(W);\gamma,T)\propto e^{-\frac{1}{T}\hat{R}(\sigma(W);C,\boldsymbol{\lambda},\gamma)},$$

$$\hat{R}\left(\sigma(W);C,\boldsymbol{\lambda},\gamma\right)=\sum_{s=1}^{S}\hat{r}\left(\sigma(\boldsymbol{w}^{(s)});C,\boldsymbol{\lambda},\gamma\right)-S\alpha\sum_{i=1}^{N}\mathrm{STD}[\{\sigma(w_i^{(s)})\}_{s=1}^{S}].$$

$\mathrm{STD}[\cdot]$: the empirical standard deviation; $\alpha\in[0,1]$: a parameter that controls solution diversity.

Experimental Setup

Implementation: PQQA is performed under four configurations: parallel runs with $S=100$ or $S=1000$, and fewer steps (3,000 steps) or more steps (30,000 steps), following the setting [2].
Metric: ApR is defined as $\mathrm{ApR}=f(\boldsymbol{x};C)/f(\boldsymbol{x}^*;C)$, where $\boldsymbol{x}^*$ denotes the optimal solution, or an approximate one obtained from best-effort optimization or asymptotic theoretical results.

Experiments

Although PQQA is evaluated on MIS, MaxClique, MaxCut, Balanced graph partition and Coloring, we focus here on summarizing the results for **MIS** as a representative example. Similar qualitative improvements are observed for the other benchmarks.

SATLIB and Erdős–Rényi Graphs.

Table 1 indicate that PQQA achieves better speed-quality trade-offs than learning-based methods, iSCO and KaMIS. For larger ER graphs, PQQA finds significantly better solutions in less time.

Table 1. The ApRs are evaluated against the results obtained by KaMIS. The baselines include OR solvers, RL-based solvers (RL), Greedy decoding (G) or Sampling (S), and iSCO.

Method	Type	SATLIB		ER-[700-800]		ER-[9000-11000]	
		ApR↑	Time↓	ApR↑	Time↓	ApR↑	Time↓
KaMIS	OR	1.000	4.50s/g	1.000	24.44s/g	1.000	28.5m/g
Gurobi	OR	1.000	3.12s/g	0.922	23.44s/g	N/A	N/A
DIMES	RL+G	0.989	2.90s/g	0.852	2.87s/g	0.841	19.54s/g
	RL+S	0.994	2.43s/g	0.937	5.63s/g	0.873	46.91s/g
iSCO	fewer steps	0.995	0.70s/g	0.998	0.65s/g	0.990	35.18s/g
	more steps	**0.996**	1.83s/g	1.006	2.61s/g	1.008	4.69m/g
PQQA	fewer ($S=100$)	0.993	0.88s/g	1.004	0.35s/g	1.027	21.86s/g
	more ($S=100$)	0.994	8.71s/g	1.005	3.33s/g	1.039	3.66m/g
	fewer ($S=1{,}000$)	**0.996**	9.00s/g	1.007	3.20s/g	1.033	2.58m/g
	more ($S=1{,}000$)	**0.996**	1.50m/g	**1.009**	32.06s/g	**1.043**	25.50m/g

Regular Random Graph with a degree $d=100$
Previous studies have highlighted the difficulty of solving the instances using learning-based solvers. Table 2 shows that QQA performs well on complex, large-scale instances. **The performance gap widens with increasing instance size, underscoring superior scalability.**

Table 2. ApR is measured with respect to the asymptotic theoretical result. PQQA is set as $S=1$.

# nodes		10^4	10^5	10^6
GREEDY		0.666 (0.02s/g)	0.667 (3.51s/g)	0.664 (5.16m/g)
SA	fewer	0.894 (3.00m/g)	0.719 (3.00m/g)	0.190 (3.00m/g)
	more	0.926 (30.00m/g)	0.887 (30.00m/g)	0.194 (30.00m/g)
iSCO	fewer	0.841 (2.04s/g)	0.781 (10.62s/g)	0.660 (1.36m/g)
	more	0.916 (11.52s/g)	0.884 (1.60m/g)	0.850 (13.99m/g)
PQQA	fewer	0.946 (1.34s/g)	0.955 (2.36s/g)	0.956 (18.47s/g)
	more	**0.957** (3.70s/g)	**0.966** (15.69s/g)	**0.966** (2.94m/g)

References

[1] Yuma Ichikawa.
Controlling continuous relaxation for combinatorial optimization.
arXiv preprint arXiv:2309.16965, 2023.

[2] Haoran Sun, Katayoon Goshvadi, Azade Nova, Dale Schuurmans, and Hanjun Dai.
Revisiting sampling for combinatorial optimization.
In *International Conference on Machine Learning*, pages 32859–32874. PMLR, 2023.

ichikawa.yuma@fujitsu.com