QUANTUM ANNEALING AND ITS VARIANTS: APPLICATION TO QUBO

13.05.2025 I VRINDA MEHTA

JÜLICH Forschungszentrum

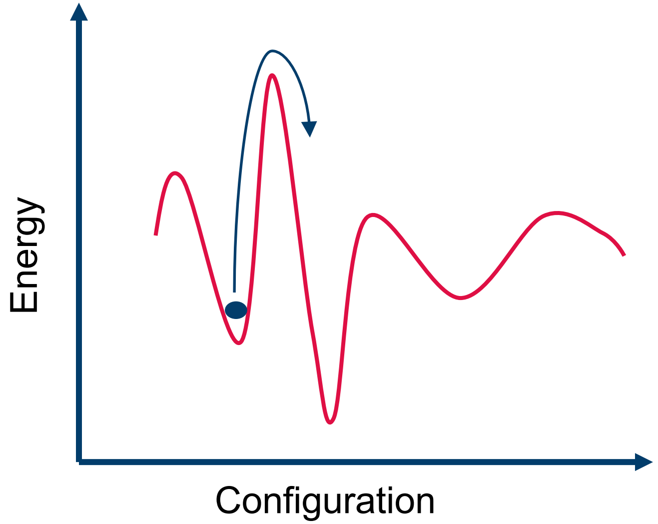
I. QUANTUM ANNEALING

1.1 Motivation

- Optimization problem: Cost function of N variables needs to be optimized
- Physically inspired techniques for solving → Map problem to a Hamiltonian

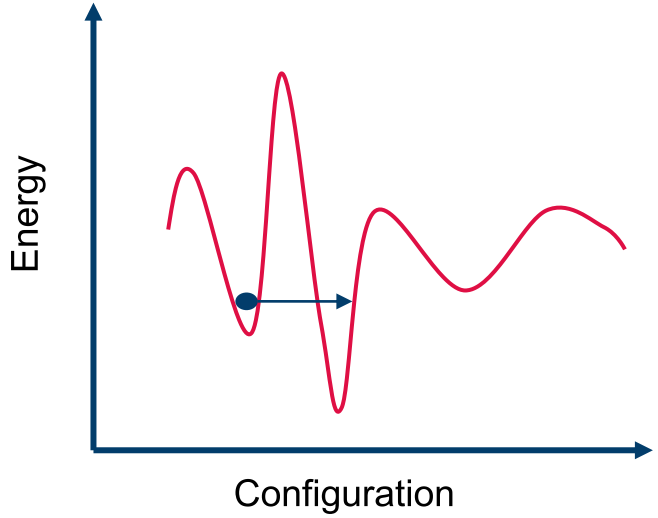
Simulated Annealing

Thermal fluctuations



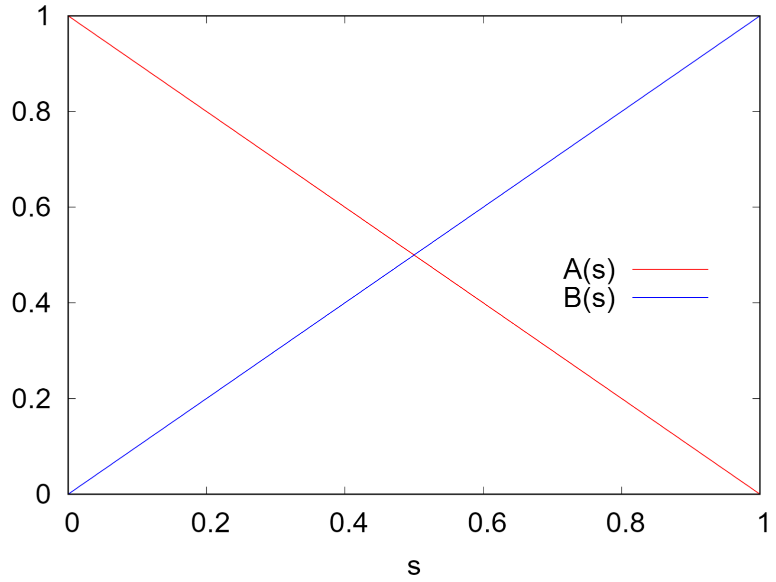
Quantum Annealing

Quantum fluctuations



I. QUANTUM ANNEALING

1.2 Annealing Hamiltonians



- Standard Hamiltonian:

$$H(s) = A(s)\sum_{i=1}^{N} h_i^x \sigma_i^x + B(s)\left(\sum_{i=1}^{N} h_i^z \sigma_i^z - \sum_{<i,j>} J_{i,j}^z \sigma_i^z \sigma_j^z\right)$$

- General Hamiltonian:

$$H(s) = \sum_{\alpha=x,y,z} f_\alpha(s)\sum_{i=1}^{N} h_i^\alpha \sigma_i^\alpha + \sum_{\alpha=x,y,z} f_{\alpha\alpha}(s)\sum_{i=1}^{N} J_{i,j}^\alpha \sigma_i^\alpha \sigma_j^\alpha$$

$s = t/T_A$
T_A = Total annealing time

- Basic ingredients
  - $H_P = -\sum_{i=1}^{N} h_i^z \sigma_i^z - \sum_{<i,j>} J_{i,j}^z \sigma_i^z \sigma_j^z$: encodes the optimization problem
  - Rest controls the choice for the initial Hamiltonian H_I and the annealing path

II. TIME-DEPENDENT SCHRÖDINGER EQUATION

- The time evolution of a quantum system is governed by the time-dependent Schrödinger equation (TDSE)

$$i\hbar\frac{\partial}{\partial t}|\psi(t)\rangle = H(t)|\psi(t)\rangle$$

$H(t)$ → Time-dependent Hamiltonian

$|\psi(t)\rangle$ → State of the quantum system at time t

$$|\psi(t)\rangle = \sum_{i=0}^{D} c_i|i\rangle$$

c_i → complex coefficients

$|i\rangle$ → basis states

- Solution: $|\psi(t+t')\rangle = U(t+t',t)|\psi(t)\rangle$
- **Time stepping**
  - $U(t+t',t) = U(t_m, t_{m-1})U(t_{m-1}, t_{m-2})\ldots.\ U(t_1, t_0)$

    where $t_j = t + j\tau$, $\tau = t'/m$ and $j: 0 \to m$
  - Approximating H to be piecewise constant, i.e., to stay relatively constant over the interval $[t_{j-1}, t_j)$

$$U(t_j, t_{j-1}) \approx e^{-i\tau H(t_{j-1}+\tau/2)}$$

III. EXACT DIAGONALIZATION

$$\psi(t+\tau) = e^{-i\tau H(t_{j-1}+\tau/2)}\psi(t)$$

- Exact diagonalization

$$H = UDU^{\dagger}$$

- Computer resources for exact diagonalization
  - Number of operations: $O(L^3)$
  - Memory: $O(L^2)$

$$L = 2^N$$

- Implementing time evolution: $e^{-i\tau H} = UU^{\dagger}e^{-i\tau H}UU^{\dagger} = Ue^{-i\tau D}U^{\dagger}$

- Number of operations for time evolution:

$$\psi(t+\tau) = e^{-i\tau H}\psi(t) = Ue^{-i\tau D}\boxed{U^{\dagger}\psi(t)} \qquad O(L^2)$$

$$= U\boxed{e^{-i\tau D}\psi'(t)} \qquad O(L)$$

$$= \boxed{U\psi''(t)} \qquad O(L^2)$$

IV. SUZUKI-TROTTER PRODUCT FORMULA ALGORITHM

4.1 Motivation

$$\psi(t+\tau) = e^{-i\tau H(t_{j-1}+\tau/2)}\psi(t)$$

- Decompose H into $H(t_{j-1}+\frac{\tau}{2}) = \sum_{k=1}^{K} H_k(t_{j-1}+\frac{\tau}{2})$

- Second-order Suzuki-Trotter product formula approximation to

$$e^{-i\tau H(t_{j-1}+\tau/2)} \approx \widetilde{U_2}\left(t_{j-1}+\frac{\tau}{2}\right)$$

$$\widetilde{U_2}\left(t_{j-1}+\frac{\tau}{2}\right) := e^{-\frac{i\tau H_K\left(t_{j-1}+\frac{\tau}{2}\right)}{2}} \dots . e^{-\frac{i\tau H_1\left(t_{j-1}+\frac{\tau}{2}\right)}{2}}\, e^{-\frac{i\tau H_1\left(t_{j-1}+\frac{\tau}{2}\right)}{2}} \dots .\, e^{-\frac{i\tau H_K\left(t_{j-1}+\frac{\tau}{2}\right)}{2}}$$

IV. SUZUKI-TROTTER PRODUCT FORMULA ALGORITHM

4.2 Example

- For

$$H(t) = -\sum_{i=1}^{N}\sum_{\alpha=x,y,z} h_i^\alpha(t)\sigma_i^\alpha - \sum_{i,j=1}^{N}\sum_{\alpha=x,y,z} J_{ij}^\alpha(t)\,\sigma_i^\alpha\sigma_j^\alpha$$

- One possible decomposition

$$H(t) = H_x(t) + H_y(t) + H_z(t)$$

where $H_\alpha(t) = -\sum_{i=1}^{N} h_i^\alpha(t)\,\sigma_i^\alpha - \sum_{i,j=1}^{N} J_{ij}^\alpha(t)\sigma_i^\alpha\sigma_j^\alpha$ for $\alpha = x, y, z$

so that

$$e^{-i\tau H_\alpha(t)} = \prod_{i=1}^{N} e^{-\tau h_i^\alpha(t)\sigma_i^\alpha} \prod_{i,j=1}^{N} e^{-\tau J_{ij}^\alpha(t)\sigma_i^\alpha\sigma_j^\alpha}$$

Single-qubit gate   Two-qubit gate

V. 2-SATISFIABILITY (SAT) PROBLEMS

- Consist of cost function of *N* Boolean variables, x_i

$$F = (L_{1,1} \vee L_{1,2}) \wedge (L_{2,1} \vee L_{2,2}) \wedge \ldots \wedge (L_{M,1} \vee L_{M,2}),$$

where $L_{\alpha,k} = x_{i[\alpha,k]}$ or $\bar{x}_{i[\alpha,k]}$ with $x_{i[\alpha,k]} = 0,1$

→ Find assignment to x_i that makes *F* true

- Reformulation as Ising Hamiltonian:

$H_{2SAT} = \sum_{\alpha=1}^{M} h_{2SAT}(\epsilon_{\alpha,1} s_{i[\alpha,1]}, \epsilon_{\alpha,2} s_{i[\alpha,2]})$, where $\epsilon_\alpha = 1$ for x_i , $\epsilon_\alpha = -1$ for $\bar{x_i}$

  - Example: for clause $(x_1 \vee x_2)$, $h_{2SAT} = s_1 s_2 - (s_1 + s_2) + 1$

- Chosen problems ($6 \leq N \leq 20$):
  - Set of 100 or 1000 problems
  - Dimension of the Hilbert space 2^N (corresponding to ≈1 000 000 complex coefficients for N=20)
  - *M* = *N*+1

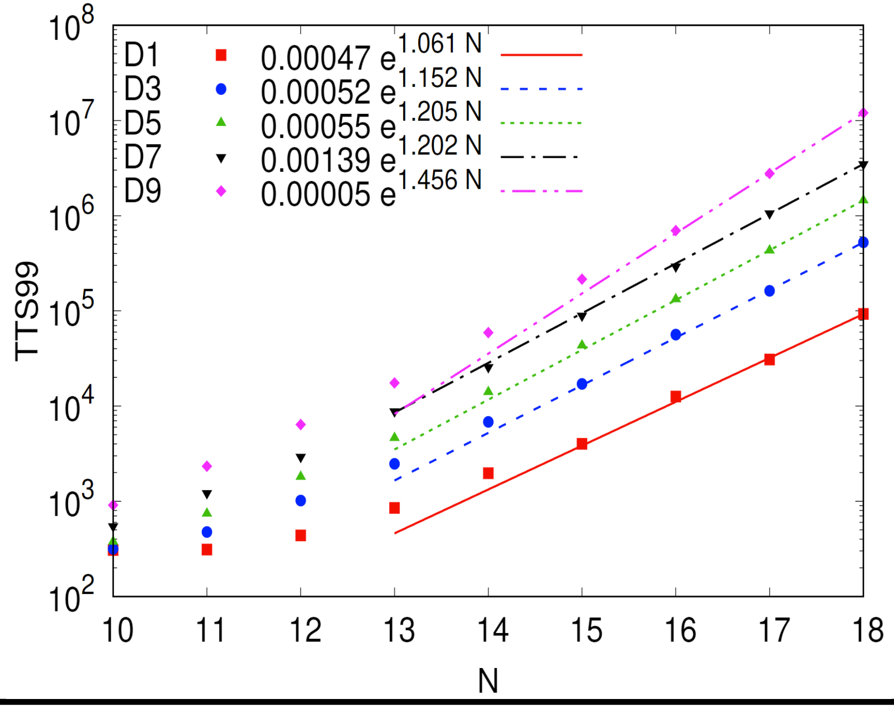
VI. STANDARD QUANTUM ANNEALING

$$\tau_{QA} = \frac{\ln(1 - P_{target})\, T_A}{\ln(1 - P_{succ}(T_A))}$$

TTS99 = τ_{QA} ($P_{target} = 0.99$)

Scaling of TTS99

Simulation, T_A = 1000



- Brute-force search scales as $e^{-0.693\,N}$

D5	:	1.205
T_A (theory)	:	1.058
Reason	:	Sufficiently long T_A

Phys. Rev. A **105**, 062406

VII. MODIFIED QUANTUM ANNEALING

7.1 Addition of Trigger Hamiltonian: Description

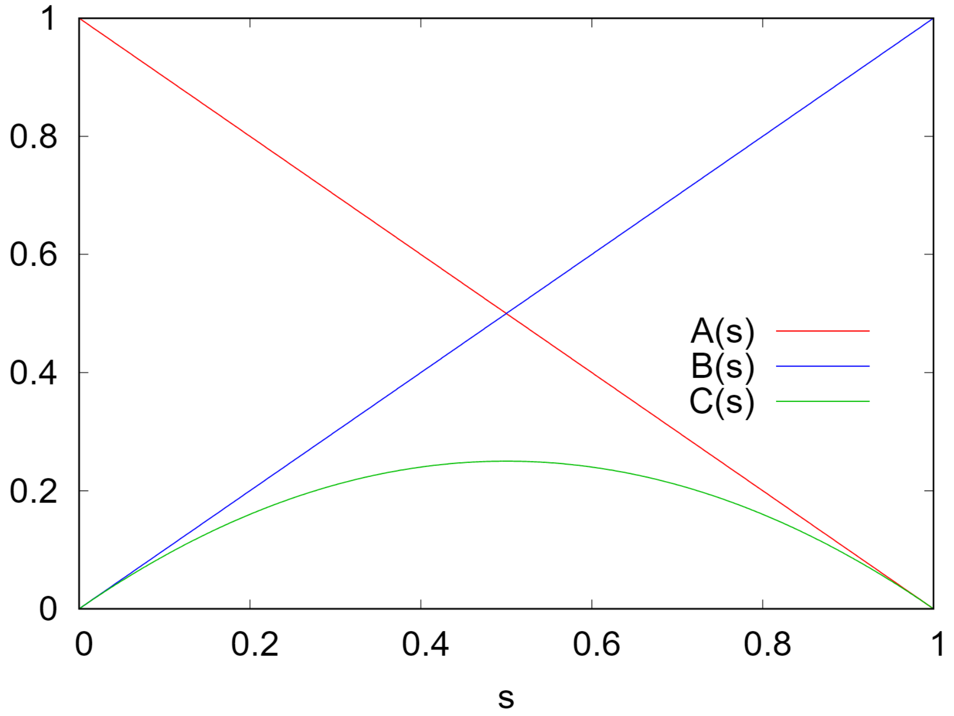
$$H(s) = A(s)H_I + B(s)H_P + C(s)H_T$$

$$H_T = -g \sum_{\{i,j\}} J_{i,j}^x \sigma_i^x \sigma_j^x$$

where,

g = strength of the trigger (0.5,1.0,2.0)

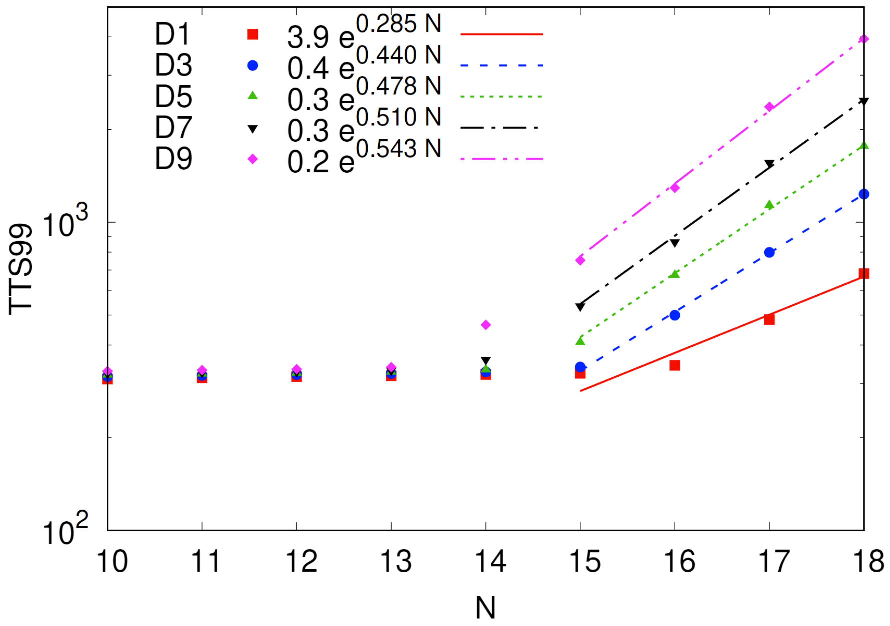
$$J_{i,j}^x = \begin{cases} 1 & \text{Ferromagnetic trigger Hamiltonian} \\ -1 & \text{Antiferromagnetic trigger Hamiltonian} \end{cases}$$



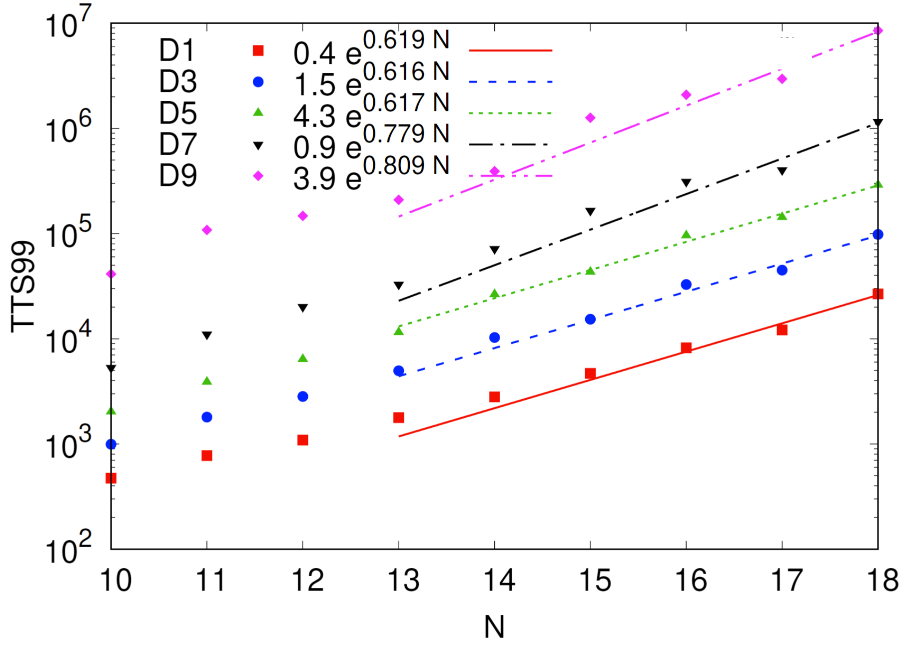
VII. MODIFIED QUANTUM ANNEALING

7.2 Scaling of TTS99

Ferromagnetic, $T_A = 1000$



Antiferromagnetic, $T_A = 1000$



Scaling improvement compared to

- Brute-force search
- Standard QA Hamiltonian

D5	**:**	0.478	0.617
T_A **(theory)**	**:**	0.424	0.530

Standard QA: 1.205

Phys. Rev. A **105**, 062406

JÜLICH Forschungszentrum

VIII. CONCLUSION

- Quantum annealing is a heuristic for solving optimization problems
- To numerically simulate → solve the time-dependent Schrödinger equation
- Numerical implementation
    - If the system size allows for it, exact diagonalization is the most straightforward method
    - If not, Suzuki-Trotter product formula algorithm is a good choice