An In-memory Computing Accelerator, CMOS Annealing Machine, to Solve Combinatorial Optimization Problems

Masanao Yamaoka
Research & Development group, Hitachi, Ltd., Tokyo, Japan

07/11/2019
MPSoC Forum 2019
Outline

• Motivations
• Overview of CMOS annealing machine
• Prototypes of CMOS annealing machine
• Related necessary technologies
• Conclusion
System of IoT era

- Collecting and analyzing a lot of data
- Results used for controlling systems
Importance of optimization

- Optimization processing used in various industries
- Optimization problem necessary to acquire optimum parameters

Logistics operation

VLSI design

Medical diagnosis with images

Management strategy

Route selection (logistics/services)

Scheduling at wide-scale disaster

Learning plan customizing

Smart-grid control

Robotics control
Combinatorial optimization problem

- Problem to explore an optimum solution for minimum KPI in given conditions
- Enormous candidates of solution with large number of parameters: $2^n$ patterns
Solving combinatorial optimization problem

- Conventional computing calculating all KPIs and deciding combination of parameters for the best KPI
- Enormous calculation required when $n$ is large
- Practically approximation algorithms used

\[ 2^n \text{ times} \]

Problem

Changing parameters
Calculating KPI

Evaluation of KPI

Exact solution
New-paradigm, natural, computing

- Mapping problems to natural phenomena
- Solution acquired by convergence operation of natural phenomena
- Ising model used for optimization problems

Conventional computing (CPU) → Using programs
Solving with procedure

Natural computing

Mapping to natural phenomena → Input
Convergence operation (annealing) → Observing
Approximation solutions

Ising model for optimization problems
Ising model

- Ising model: expressing behavior of magnetic spins, upper or lower directions
- Spin status updated by interaction between spins to minimize system energy

\[ H = -\sum_{\langle i,j \rangle} J_{ij}\sigma_i\sigma_j - \sum_j h_j\sigma_j \]

- \( H \): Energy of Ising model
- \( \sigma_i \): Spin status (+1/-1)
- \( J_{ij} \): Interaction coefficient
- \( h_j \): External magnetic coefficient
Computing with Ising model: annealing machine

- Shape of landscape of Ising model energy same as KPI plot of combinatorial optimization problem
- By mapping original problems to Ising model, optimum solution acquired as ground state of model

\[ H = - \sum_{\langle i,j \rangle} J_{ij} \sigma_i \sigma_j - \sum_j h_j \sigma_j \]

Correspondence of parameters:

<table>
<thead>
<tr>
<th>Ising model</th>
<th>Optimization problems</th>
</tr>
</thead>
<tbody>
<tr>
<td>Energy ( H )</td>
<td>KPI</td>
</tr>
<tr>
<td>Spin status ( \sigma_i )</td>
<td>Control parameters</td>
</tr>
<tr>
<td>Interaction coefficient ( J_{ij} )</td>
<td>Input data (sensor data, etc)</td>
</tr>
</tbody>
</table>

Energy of system \( H \)

(KPI)

Spin status (\( 2^n \) patterns)

\( n \): number of spins

Ground state
Natural computing using Ising model: annealing machine

- Mapping optimization problems to Ising model
- Convergence operation of Ising model
- Solution acquired by observing convergence results

Problem

Problem mapping to Ising model

Convergence operation of Ising model

Observation of results

Solution
Various implementation of annealing machines

- Quantum annealing machine, coherent Ising machine, CMOS annealing machine are proposed
- All machines based on Ising model

<table>
<thead>
<tr>
<th></th>
<th>Quantum annealing machine</th>
<th>Coherent Ising machine</th>
<th>CMOS annealing machine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Principle</td>
<td>Quantum annealing</td>
<td>Parametron</td>
<td>Classical annealing</td>
</tr>
<tr>
<td>Implementation</td>
<td>Superconductor</td>
<td>Laser oscillator</td>
<td>CMOS</td>
</tr>
<tr>
<td>Power</td>
<td>Large for cooling</td>
<td>Good</td>
<td>Better</td>
</tr>
<tr>
<td>Topology</td>
<td>Sparse</td>
<td>All to all</td>
<td>Sparse</td>
</tr>
</tbody>
</table>
Comparison of annealing machines

- For edge: Low cost, real time, room temperature, low power
- For cloud: High speed, low power, large scale, expandability

<table>
<thead>
<tr>
<th>Method</th>
<th>Conventional computer</th>
<th>Quantum annealing machine</th>
<th>CMOS annealing machine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Devices</td>
<td>CPU</td>
<td>Superconductor</td>
<td>Semiconductor (CMOS)</td>
</tr>
<tr>
<td>Approach</td>
<td>Digital</td>
<td>Quantum bit</td>
<td>Digital</td>
</tr>
<tr>
<td>Calculation time</td>
<td>×</td>
<td>◎</td>
<td>○</td>
</tr>
<tr>
<td>Temperature</td>
<td>Room temp.</td>
<td>15mK</td>
<td>Room temp.</td>
</tr>
<tr>
<td>Operation power</td>
<td>10-1,000W</td>
<td>15,000W (with cooling)</td>
<td>0.05W</td>
</tr>
<tr>
<td>Scalability</td>
<td>Enable</td>
<td>512 ('12) → 2048 ('17)</td>
<td>100kbit ('18)</td>
</tr>
<tr>
<td>Expandability</td>
<td>Enable</td>
<td>-</td>
<td>Multiple-chip</td>
</tr>
</tbody>
</table>
Outline

- Motivations
- Overview of CMOS annealing machine
- Prototypes of CMOS annealing machine
- Related necessary technologies
- Conclusion
In-memory computing for Ising-model computing

- In-memory structure suitable for mimicking Ising model convergence operation
- No memory access required in convergence operation

Problem:

1. Problem mapping to Ising model
2. Input data into in-memory structure

Solution:

1. Ising model convergence operation within in-memory structure
2. Output of results
In-memory computing for Ising model

- Mimicking physical Ising model with in-memory structure
- Spin status updated by logic circuits implemented in memory

<table>
<thead>
<tr>
<th>Ising model</th>
<th>CMOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spin status $\sigma_i$, +1/-1</td>
<td>Memory &quot;1&quot;/&quot;0&quot;</td>
</tr>
<tr>
<td>Interaction $J_{ij}$</td>
<td>Interaction coefficient: memory</td>
</tr>
<tr>
<td></td>
<td>Interaction operation: logic circuits</td>
</tr>
</tbody>
</table>

CMOS circuit mimicking Ising model
CMOS annealing

- Only digital operation, spin status stuck at local minimum status
- To avoid local minimum sticking, random status transition used
- Optimum solution not always acquired

Energy of system $H$ (KPI)

Spin status ($2^n$ patterns)

$n$: number of spins

- Transition to lower energy (adjacent spin interaction)
- Avoidance of local minimum (random transition)
Rule of spin status update

- Spin status updated to lower Ising model energy
- Coefficient +: same direction
  Coefficient -: opposite direction
- Majority of adjacent spins effect accepted

\[ H = - \sum_{(i,j)} J_{ij} \sigma_i \sigma_j - \sum_j h_j \sigma_j \]

Spin status:
\( J_{ij} > 0 \): same direction
\( J_{ij} < 0 \): opposite direction

Next spin status:
- in case \( a>b \), \( \sigma_5 = +1 \)
- in case \( a<b \), \( \sigma_5 = -1 \)
- in case \( a=b \), \( \sigma_5 = +1 \) or \( -1 \)

\( a=\) number of \((+1, +1)\) or \((-1, -1)\)
\( b=\) number of \((+1, -1)\) or \((-1, +1)\)
(value from adjacent spin, coefficient)
In-memory structure of CMOS annealing machine

- Spin-update rule achieved by majority voter
- Each spin has update circuit and updated in parallel
- Short calculation time even when number of spins is large

CMOS annealing machine

Spin unit

Interaction calculation

SRAM Digital circuits
Circuit structure for 1 spin unit

- Next spin status digitally calculated
- Majority voter circuits for efficient calculation

\[ H = -\sum_{<i,j>} J_{ij}\sigma_i\sigma_j - \sum_j h_j\sigma_j \]

**Spin update rules**

Next spin status:
- in case \( a > b \), \( \sigma_5 = +1 \)
- in case \( a < b \), \( \sigma_5 = -1 \)
- in case \( a = b \), \( \sigma_5 = +1 \) or \(-1\)

\( a \) = number of \((+1, +1)\) or \((-1, -1)\)
\( b \) = number of \((+1, -1)\) or \((-1, +1)\)
(value from adjacent spin, coefficient)
EXOR and majority voter circuit

- EXOR and sum of results of EXOR ("0" / "1") calculated by sum of currents.
- Voltage of common lines evaluated by sense amplifiers.

Precharge CT for numbers of (adjacent spin value, coefficient)= (+1, +1) or (-1, -1)

CB with same structure circuits for numbers of (-1, +1) or (-1, +1)
• Motivations
• Overview of CMOS annealing machine
• Prototypes of CMOS annealing machine
• Related necessary technologies
• Conclusion
Fabrication results: CMOS annealing chip

<table>
<thead>
<tr>
<th>Items</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of spins</td>
<td>20k (80 x 128 x 2)</td>
</tr>
<tr>
<td>Process</td>
<td>65 nm</td>
</tr>
<tr>
<td>Chip area</td>
<td>4x3=12 mm$^2$</td>
</tr>
<tr>
<td>Number of SRAM cells</td>
<td>260k bits</td>
</tr>
<tr>
<td></td>
<td>Spin value: 20k bits</td>
</tr>
<tr>
<td></td>
<td>Interaction coefficient: 240k bits</td>
</tr>
<tr>
<td>Memory IF</td>
<td>100 MHz</td>
</tr>
<tr>
<td>Interaction speed</td>
<td>100 MHz</td>
</tr>
<tr>
<td>Operating current of core circuits</td>
<td>Write: 2.0 mA</td>
</tr>
<tr>
<td></td>
<td>Read: 6.0 mA</td>
</tr>
<tr>
<td></td>
<td>Interaction: 44.6 mA</td>
</tr>
</tbody>
</table>
1st generation prototype

- Ising chips installed on computing node
- FPGA installed to control Ising chip
- Accessed via LAN cable from PCs/servers

### Computing node

<table>
<thead>
<tr>
<th>Configuration</th>
<th>2U rack mount</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation frequency</td>
<td>100 MHz</td>
</tr>
<tr>
<td>Number of spins</td>
<td>40k (2chips)</td>
</tr>
<tr>
<td>OS</td>
<td>Linux</td>
</tr>
</tbody>
</table>
Random transition: random numbers

• Two sequences of 1-bit random numbers input, propagated, and evaluated at spin interaction
• Only two PRNGs provide randomness to whole chip

PRNG: Pseudo Random Number Generator

Zigzag paths

Spin inversion using random pulse

➢ Output of the interaction circuit will be inverted if both of the random pulses are ‘1’

Spin

XOR

AND

Majority voter

Spin unit

Random pulse 1

Random pulse 2
Random pulse control

- Spin flip rate gradually lowered for annealing
- Spin flip rate controlled by mark ratio of 1bit PRNGs

Controlling mark ratio of 1bit PRNG

1bit PRNG

\[ \begin{array}{c}
\text{PRNG} \\
0 \quad 1
\end{array} \]

r

Comparator

0 (r < Threshold)

1 (Otherwise)

e.g. LFSR

Threshold

Mark ratio during ground-state search

Aggressively escape from local minima

Settle to nearby low energy solution

Mark ratio

Expected spin flip rate

Time

Decreasing mark ratio has similar effect to cooling schedule in Simulated Annealing

Time
Measurement results of CMOS annealing

- MAX-cut problem, NP-complete problem, with 20k-spin solved
- Coefficient values set "ABC" appeared at optimum
- Optimum solution not always acquired

![Graph showing energy over time](image_url)

(a) Initial state
(b) 5 ms (500,000 steps)
(c) 10 ms (1,000,000 steps)
Operating energy

- 1,800 times higher energy efficiency for 20k spin problem than approximation algorithm on CPU

<table>
<thead>
<tr>
<th>Number of spins (problem size)</th>
<th>Energy efficiency vs approximate algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>64</td>
<td>10</td>
</tr>
<tr>
<td>512</td>
<td>100</td>
</tr>
<tr>
<td>4096</td>
<td>1000</td>
</tr>
<tr>
<td>32768</td>
<td>10000</td>
</tr>
</tbody>
</table>

Conditions: Randomly generated Maximum-cut problems, energy for same accuracy solution Ising chip: VDD=1.1 V, 100-MHz interaction, best solution among 10-times trial is selected. Approximation algorithm: SG3(*) is operated on Core i5, 1.87 GHz, 10 W/core.
2nd generation prototype

- For software development, FPGA prototype used
- Various structures for trials (various topology, various bit number of coefficient)
Scalable inter-FPGA connectivity

- Operate as a large-scale machine using chip-to-chip connection
- Local transmission for scalable connection of many FPGA boards
2nd generation 100kbit prototype

• Connect 5x5 FPGAs and demonstrate 100k bit operation with the largest number of parameters
### 100kbit CMOS annealing machine

<table>
<thead>
<tr>
<th>Item</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of parameters</td>
<td>102,400</td>
</tr>
<tr>
<td>Connection of Ising model</td>
<td>Partially coupled</td>
</tr>
<tr>
<td>Number of FPGA boards</td>
<td>25 (4,096 per 1 FPGA)</td>
</tr>
<tr>
<td>FPGA operating frequency</td>
<td>82.5MHz</td>
</tr>
<tr>
<td>Parameter resolution</td>
<td>5bit (±15)</td>
</tr>
<tr>
<td>FPGA board</td>
<td>Xilinx® UltraScale®</td>
</tr>
<tr>
<td>Interconnect of FPGA</td>
<td>Xilinx® Aurora®</td>
</tr>
</tbody>
</table>
Performance comparison

- 156x speed improvement compared to conventional computer
- Larger number of parameters, greater speed improvement

Speed comparison with conventional machines

Evaluation condition

Execute ground state search of randomly generated Ising model with SA on CMOS annealing machine and conventional computer (CPU), and compare time to reach reference accuracy obtained by existing algorithm.
Annealing technique for higher precision

- Map of annealing machine and algorithms

- Optoelectronic silicon chip
- CMOS annealing machine
- Coherent Ising machine
- Quantum annealing machine
- New CMOS annealing machine with SQA
- Software SA
- Software SQA
- SQA: Simulated Quantum Annealing
3rd generation prototype

- Include SQA algorithm to improve performance
- Improve solution accuracy by incorporating pseudo quantum effects

<table>
<thead>
<tr>
<th>Item</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithm</td>
<td>Simulated Quantum Annealing (SQA)</td>
</tr>
<tr>
<td>Implementation</td>
<td>FPGA</td>
</tr>
<tr>
<td>Topology</td>
<td>King graph</td>
</tr>
<tr>
<td># of spins</td>
<td>2500 (50×50)</td>
</tr>
<tr>
<td># of replicas</td>
<td>32</td>
</tr>
<tr>
<td>Coefficient</td>
<td>8 bits (0, ±1, ..., ±127)</td>
</tr>
<tr>
<td>Interaction</td>
<td>50 MHz</td>
</tr>
</tbody>
</table>
Simulated annealing

- SA consists of discrete-time Markov processes that converge to Boltzmann distribution

Hamiltonian for Simulated Annealing

\[ H_{SA} = - \sum_{(i,j)} J_{ij} \sigma_i \sigma_j - \sum_i h_i \sigma_i \]
Simulated Quantum Annealing

- Reproduce the quantum superposition effect by the SQA method
- Digital circuit implements SQA method

Hamiltonian for Simulated Quantum Annealing

\[ H_{\text{SQA}} = - \sum_{k=1}^{M} \left( \sum_{(i,j)} \frac{J_{ij}}{M} \sigma_{i,k} \sigma_{j,k} + \sum_{i} \frac{h_i}{M} \sigma_{i,k} + J^+ \sum_{i} \sigma_{i,k} \sigma_{i,k+1} \right) \]

Strength of coupling between replicas \( J^+ \)

1\(^{st}\) replica
2\(^{nd}\) replica
\vdots
\( M^{th}\) replica
CMOS annealing machine based on SQA

- Each replica has the same combination of couplings and biases
- Memory for interaction coefficients shared
Speed comparison for optimization problem calculation

- Executes optimization processing (SA) 40 times faster than conventional methods

Note:
- Total annealing time means a time to obtain 99.9% solution with a probability of 99%
- 200 different random Ising models on a king graph are used
- We run the optimized SA program on a state-of-the-art CPU (Intel Core i7-6700K, 8 threads)
Performance of SQA based CMOS annealing machine

- Supremacy over software SA for large size problems
- Computational amount reduced:
  Contributing to higher speed, lower power consumption

![Graph showing comparison between CIC and SA methods](image)
For larger-scale implementation

Size

1k 10k 100k

Rack

1st generation (ASIC) 2015/02 released

2nd generation (FPGA) 2016/06 released

3rd generation (FPGA) 2017/10 released

Node

Cloud

Multi-FPGA implementation

Card

Edge

Multi-ASIC implementation

Number of bits
Card-size CMOS annealing machine for edge devices

- Prototype of 30-k spin annealing chip in 40 nm CMOS process
- Card sized CMOS annealing machine equipped with 2 chips
  (Realized optimization calculation of about 60,000 parameters)
Proposed inter-chip Interface

- 100-Mb/s LVDS I/F: 2×88-bits data are split into 8×22-bits chunks
- Update values of dummy spin based on spin update rule
Annealing speed compared to CPU

- Fast annealing time: 22 μs (=22 clock cycles x 100 annealing steps)
- 2 x 30k spin system Max-Cut problem with randomly allocated coefficients

**CMOS annealing processor** **2.6×10⁴ times faster than CPU**

**Without I/O time** (23.8 ms) through USB3.0 I/F
Two-chip operation for Max-Cut problem

- 2 chip operation confirmed with max cut problem
- Border line between chips disappears in the final low-temperature state

Energy of Ising model as a function of annealing time. (i) 0 µs (initial state), (ii) 7.7 µs, (iii) 21.8 µs (final state).

One pixel is one spin.
Energy efficiency compared to CPU

- Energy efficiency improves with increasing number of spins
- 2x30k spin system: $1.75 \times 10^5$ higher than CPU

![Graph showing energy-efficiency ratio vs number of activating spins.

Energy efficiency ratio = \frac{\text{Problem size} \times \text{Calculation time}}{\text{Power consumption}}

* CPU (Intel Corei7-6700K @4.00GHz): SG3 algorithm
Comparison of annealing machines

- Multi-chip operation for larger scale confirmed
- Faster operation and higher energy efficiency achieved by new chip

<table>
<thead>
<tr>
<th>Method</th>
<th>D-Wave 2000Q</th>
<th>1st gen.</th>
<th>Multi-chip version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>Quantum annealing</td>
<td>Simulated annealing</td>
<td>Good</td>
</tr>
<tr>
<td>Implementation</td>
<td>Superconductor</td>
<td>65-nm CMOS</td>
<td>40-nm CMOS</td>
</tr>
<tr>
<td>Number of chips</td>
<td>1</td>
<td>1</td>
<td>2 (multichip in principle)</td>
</tr>
<tr>
<td>Number of spins</td>
<td>2k</td>
<td>20k</td>
<td>2 × 30k</td>
</tr>
<tr>
<td>Annealing time*</td>
<td>-</td>
<td>10 ms***</td>
<td>22 μs</td>
</tr>
<tr>
<td>Energy efficiency*, **</td>
<td>-</td>
<td>2200 times***</td>
<td>1.75 × 10⁵ times</td>
</tr>
</tbody>
</table>

* Max-Cut problem is applied. ** These values are evaluated by comparing against running SA on CPU. *** These values are evaluated under the condition similar to this work.
Benefits vs a traditional computation

- **In-memory implementation for Ising model computing:**
  Lower power operation with local-area process
  Higher operation speed with parallel operation

- **Quantum-inspired algorithm:**
  Higher speed and lower power consumption with smaller computational amount

- **Multi-chip implementation by in-memory structure:**
  Large-scale integration for larger problems
• Motivations
• Overview of CMOS annealing machine
• Prototypes of CMOS annealing machine
• Related necessary technologies
• Conclusion
How to solve "real" problems

<table>
<thead>
<tr>
<th>Systems</th>
<th>Supply chain</th>
<th>Finance</th>
<th>Traffic</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>Reduction of logistics cost</td>
<td>Optimizing Portfolio</td>
<td>Reduction of traffic jam</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Software</th>
<th>Problem</th>
<th>Mapping</th>
<th>Graph embedding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware</td>
<td>Problem</td>
<td>Mapping</td>
<td>Graph embedding</td>
</tr>
</tbody>
</table>

**Cost function of optimization problems**

- Max-flow problem
- Knapsack problem
- Shortest path problem

**Formulation of Ising model**

- Ising model on computer
- Data IO ($J_{ij}, h_i$)

**Hardware**

- Ising Computer

**Equations**

- $J_{12} = x$, $J_{23} = y$, ...

- Ising model: $s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9$
### How to solve "real" problems

<table>
<thead>
<tr>
<th>Systems</th>
<th>Supply chain</th>
<th>Finance</th>
<th>Traffic</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Problem mapping</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Graph embedding</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hardware</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Implementation of PoC in cooperation with users

- Reduction of traffic jam
- Reducing logistics cost
- Optimizing portfolio
- Traffic
- Finance

#### Promote open development in collaboration with universities etc.

- Ising model on computer
- Max-flow problem
- Shortest path problem
- Knapsack problem

#### Forming a community and creating a current flow for new computers

- Ising Computer
- $J_{12} = x, J_{23} = y, \ldots$
Application of CMOS annealing machine

1. Frequency allocation for wireless radio
2. Communication order allocation
3. Server security
4. Image inpainting
5. Exploring explosion material
6. Image noise reduction
7. Facility allocation
8. Community core detection
9. Machine learning (Boosting)
Operation procedure

- Pre-processing required for application-specific computing using Ising model
- Generated data easily input/output using memory IF

Cost function of optimization problems

Formulation of Ising model

Write input params \((J_{ij}, h_i)\)  

Annealing operation

Readout of spin states \((\sigma_i)\)

(Near-) optimal solution
Number Partitioning Problem (NPP)

- Finding a division of a set into 2 subsets such that the sums of each subset are as close as possible

\[ S = \{ 3, 1, 4, 15, 92, 65, 35, 89, 79, 32, 38, 46, 26, 43, 38, 32, 79, 50, 28, 84 \} \]

\[ \begin{align*}
  &1, 65, 35, 38, 46, 26, 38, 79, 28, 84 \\
  &3, 4, 15, 92, 89, 79, 32, 43, 32, 50
\end{align*} \]
Mapping of NPP to Ising model

- Formulation of NPP mapped to Ising model formulation

Original formulation

$$\min_{\sigma_1, \ldots, \sigma_n \in \pm 1} \sum_i w_i \sigma_i$$

Ising formulation of NPP

$$H(\sigma) = \sum_{i<j} w_i w_j \sigma_i \sigma_j$$

$w_i$ : value of $i$-th element of the set $S$

$\sigma_i$ : label of $i$-th element
Mapping of constrained NPP to Ising model

- Constrained optimization problem also mapped to Ising model formulation

**Original formulation**

$$\min_{\sigma_1, \ldots, \sigma_n \in \pm 1} \sum_i w_i \sigma_i$$

subject to $$\sum_i \sigma_i = 0$$

- Same number of items for each group

**Ising formulation of constrained NPP**

$$H(\sigma) = \sum_{i<j} w_i w_j \sigma_i \sigma_j + \lambda \left( \sum_i \sigma_i \right)^2$$

- $w_i$ : value of $i$-th element of the set $S$
- $\sigma_i$ : label of $i$-th element
Machine learning using Boltzmann machine

• Boosting technique and reinforcement learning proposed

Boosting: finding optimum set of weak classifiers

Reinforcement learning: using Ising model as Boltzmann machine instead of neural network
Graph embedding required for in-memory structure

- Many original problems have complex graph topology
- Graph embedding to use in-memory structure efficiently

**Diagram:**
- **Problem mapping**
  - Express by Ising model
  - Ising model (physics model)
  - Transformation algorithm
  - Transform to HW topology
  - Ising computer

**Graph embedding**

**Processing**
Cloud environment for beginners

• Access to CMOS annealing machine via internet
• Tutorials and demos to understand annealing machine

https://annealing-cloud.com/

This web page is based on results obtained from a project commissioned by the New Energy and Industrial Technology Development Organization (NEDO).
Program contest for annealing machine

- Program contest for graph embedding in 2017, and for lowering graph order in 2018
- From the junior high school students to the 50s, answer codes submitted from some countries

Result

<table>
<thead>
<tr>
<th>Радиоприемник</th>
<th>Радиомикрофон</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st (entry: 973 submitted: 296)</td>
<td>2nd (entry: 446 submitted: 126)</td>
</tr>
</tbody>
</table>

Postscript of participation

Award ceremony (information processing national convention)
Conclusion

- New paradigm, natural computing is necessary for system optimization with large amount of data.
- CMOS annealing machine for combinatorial optimization problem is proposed.
- 1st to 3rd, and multi-chip machines are developed to solve larger problems.
- Software techniques and related hardware techniques are necessary for practical application.
Acknowledgement

• Part of this work is based on results obtained from a project commissioned by the New Energy and Industrial Technology Development Organization (NEDO).