Stochastic Problem Solving by Local Computation based on Self-organization Paradigm

Yasusi Kanada
Tsukuba Research Center
Real World Computing Partnership

Masao Hirokawa
Advanced Research Laboratory
Hitachi, Ltd.

Problems of real-world computational systems

Introduction

- **Future real-world computational systems are** (such as secretary robot brains)
  - Complex
    - “Non-linear,” or
    - Undecomposable into “independent” modules (because of strong interaction between modules)
  - Open and adaptive to real world (i.e., to humans and/or natural systems)
    - Adaptive to unexpected inputs (in a short period of time) — humans and natural systems are unpredictable because autonomous and nondeterministic.
    - Adaptive to environmental change (in a long period of time)
- **In development of real-world computational systems**
  - No global and complete specifications can be written, because open to real world
  - Top-down design or divide-and-concur method do not work well, because of no complete specification, and complexity.

What is the self-organization paradigm?

- **What is self-organization?**
  - An emergent behavior — toward “global order” from local motion
  - We should learn from nature.
    - Natural sytems are self-organizing systems.
    - Natural sciences on self-organizing systems: *Dissipative structure theory* by Prigogine, *Synergetics* by Haken, *Molecular evolution theory* by Eigen, *Autopoiesis theory* by Matrana and Varela, *Bio-holonics* by Shimizu, Natural and artificial *neural networks*, ….
- **“Global order” from computation with local information**
  - Computation only with local and partial knowledge — no algorithms.
  - Computation only with partial specification! (or no specification?)
- **The knowledge shortage must be covered by**
  - Nondeterminism (trial and error, or random selections) in short range.
  - Self-organization in long range. (Nondeterminism is important for self-organization.)

Research goals

- **Long-term research goals**
  - To develop a new problem-solving methodology based on a self-organization paradigm.
  - To develop adaptive and open computational systems.
- **We are only at the beginning of research toward these goals.**
- **Short-term research objective**
  - To establish a computation mechanism and methodology, which are
    - Emergent and nondeterministic
    - Based on local and partial information.

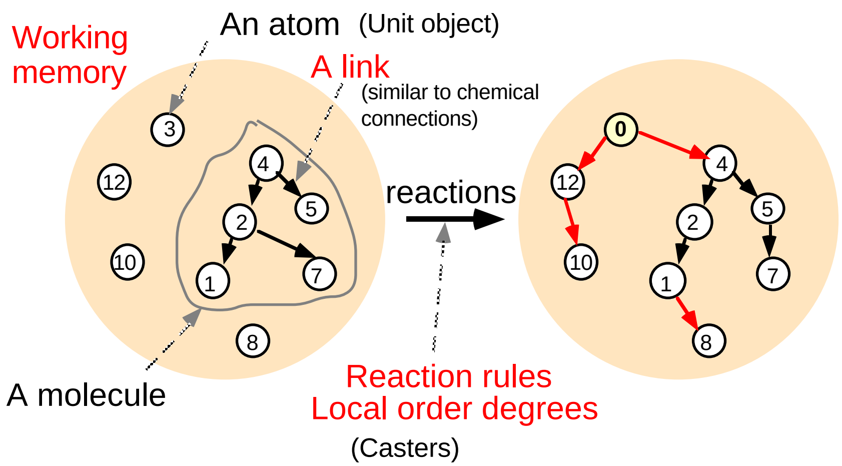
Computation model CCM

A microscopic model of computation

- **We develop a computation model called CCM for self-organizing computation.**
  - CCM is an abbreviation of “Chemical Casting Model.”
  - “Chemical” means CCM has an analogy to chemical systems.
  - “Casting” means programming or computation.
    - I do not use “program” because it means a whole and complete plan.

Components of CCM — 1

Outline



Working memory

An atom (Unit object)

A link (similar to chemical connections)

reactions

A molecule

Reaction rules
Local order degrees
(Casters)

Components of CCM — 2

Casters (Programs) of CCM

- **A caster consists of**
  - Local order degrees (LODs)
  - Reaction rules
- **LODs**
  - Are local evaluation functions (or negative energy).
    - “Local” means “defined on a small number of data.”
  - Are defined for an atom or between two or more atoms.
- **Reaction rules**
  - Change partial (local) state of the working memory.
  - Are written as forward-chaining production rules, such as
    - Chemical reaction formulae.
    - Rules in production systems, used for building expert systems.

Computation process in CCM

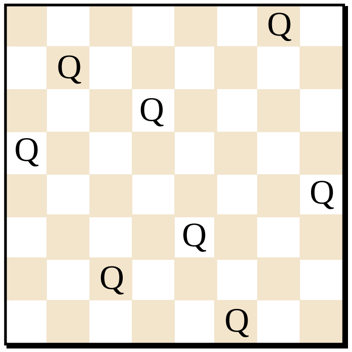
- **A reaction**
  - An application of a reaction rule is called a reaction.
  - A reaction takes place when
    - There are a rule and a set of data that match the LHS of the rule, and
    - The sum of LODs of the data, concerning the reaction, does not decrease by the reaction.
- **Succession and termination of reactions**
  - Reactions occur successively when possible.
    - Their order is nondeterministic (or random) — No limit cycles occur!
  - If no reaction can occur,
    then the system (temporarily) terminates.
  - The system may begin to work again,
    when data are modified, removed or added externally.

The *N* queens problem

Example: the *N* queens system — 1

- **The *N* queens problem**
  - An extension of the eight queens problem.
  - A problem of finding a layout of *N* queens on *N* x *N* “chess board,” where a queen does not take each other.
- **The *N* queens system**
  - A computational system to solve the *N* queens problem in CCM.
- **The reasons that we use the *N* queens problem**
  - We have to start with a simpler system.
  - This system has several characteristics that will probably lead us to a better understanding of complex systems.

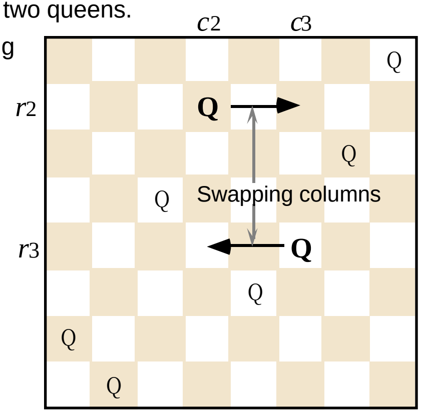
A solution of the eight queens problem



How to solve the *N* queens problem?

Example: the *N* queens system — 2

- **Using swap operations**
  - A reaction swaps the columns of two queens.
  - To solve the problem by repeating the swaps of different queens.
- **The initial conditions**
  - All the queens are put on the board from the beginning.
  - There is only one queen in each row and each column.
    - Example: all the queens can be put on a diagonal line.
    - This condition holds at any time because the reaction preserves it.



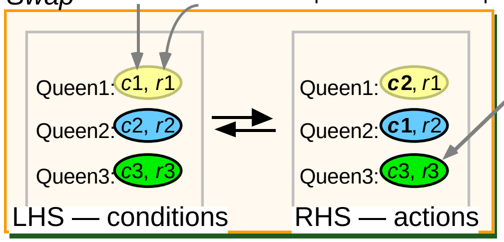
The caster for the *N* queens system

Example: the *N* queens system — 3

- **Reaction rule** (only one)

rule *Swap* column row : the representation of a queen (coordinates)

an atom



Not changed (called a *Catalyst* — explained later)

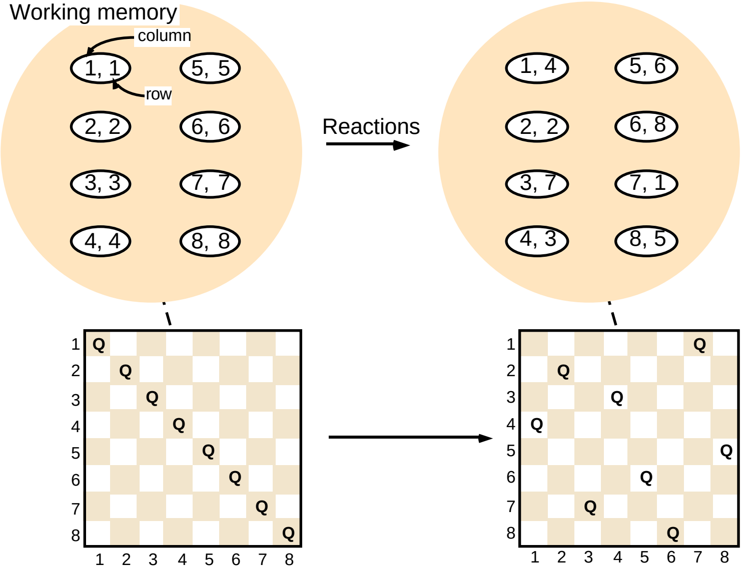
*c*1, *c*2, *c*3, *r*1, *r*2, *r*3: variables

- **Local order degree** (Mutual order degree)
  - Definition: $o(x, y) = 0$ **if** *x.column* – *y.column* = *x.row* – *y.row* **or** *x.column* – *y.column* = *y.row* – *x.row*,
    1 **otherwise**.
  - Meaning:
    If queens *x* and *y* are diagonally oriented, then 0. Otherwise, 1.

Less ordered

More ordered

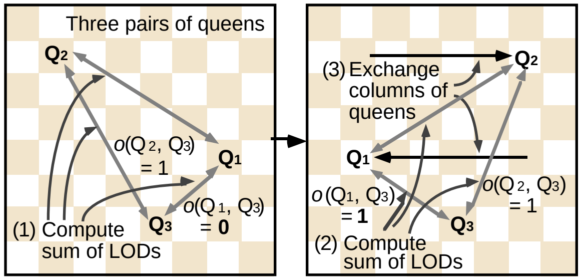
Content of working memory for the eight queens



A more detailed semantics of reactions

Example: the *N* queens system — 4

- **Selections of a rule and objects**
  - No need to select a rule because there is only one rule.
  - Three queens are *nondeterministically* (randomly) selected and reacted.
- **Computation of order degrees**
  - The sums of LODs before and after the reaction are computed (before the reaction).



- **The reason that the catalyst (Q_3) is necessary**
  - The sum of LODs is not changed if the rule contains only Q_1 and Q_2, because the LOD between Q_1 and Q_2 is not changed.
    - So the system does not stop when a solution is found.

Performance evaluation — 0*

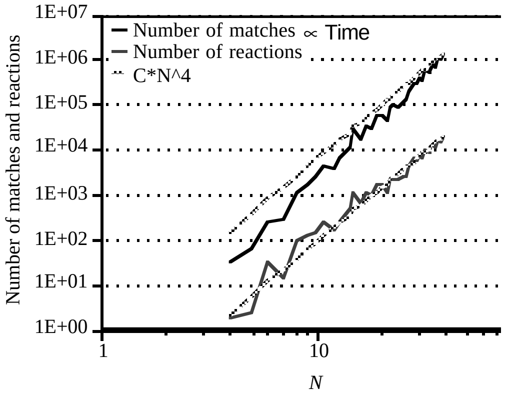
Several conditions of the measurement

- **The performance of the *N* queens system is measured using SOOC.**
  - SOOC (Self-Organization-Oriented Computing) is a computation language based on CCM.
- **The initial layouts of queens are random.**
- **All values are averages of ten executions.**

Performance evaluation

Results of the *N* queens

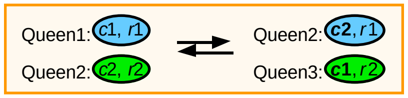
- **The problems never fail to be solved in our experiments,**
  - In spite of the stochastic and non-exhaustive search method.
- **The execution time is in polynomial order ($O(N^{4.6})$).***
  - Much faster than blind backtrack search ($O(e^N)$).
  - It is slower than more intelligent methods (Yagrom's method — $O(N)$).



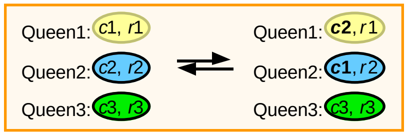
Locality control by catalysts — 1

Variability of locality

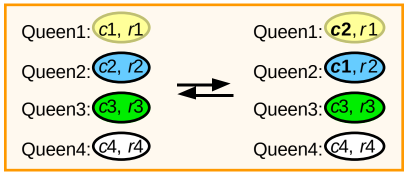
- **The locality of data reference can be controlled by adding/removing catalysts to rules.**
- **Versions of the *N* queens rule**
  - A rule with no catalyst (*Nc* = 0):
    - Most local (minimum data reference)



  - A rule with one catalyst (*Nc* = 1):



  - A rule with two catalysts (*Nc* = 2):
    - Less local

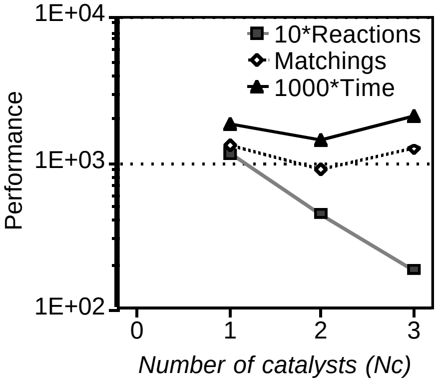


  - A rule with *N* – 2 catalysts (*Nc* = *N* – 2)
    - Global — all the queens are referred.

Locality control by catalysts — 2

Performance comparisons when changing *Nc*

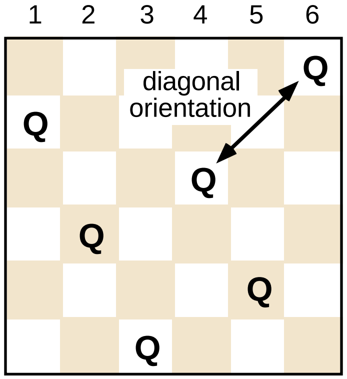
- **No catalyst**
  - The system does not stop even when a solution is found — because there is no bias toward solutions.
    - The execution time is infinite.
- **One catalyst or more**
  - The number of reactions decreases when *Nc* increases.
  - The execution time is optimum when *Nc* = 2.



Locality control by catalysts — 3

Escaping from “local maxima”

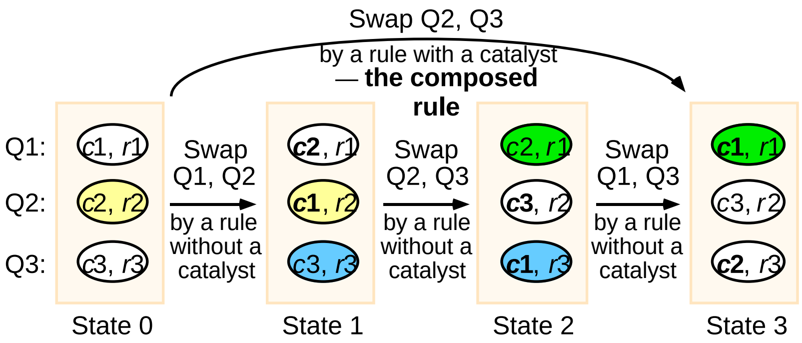
- **No catalyst**
  - No bias (complete random walk) — no local maxima (of global order degree — the total of LODs (negative total energy)).
- **One catalyst or more**
  - There may be local maxima — invalid termination.
  - If less catalysts — less chances to fall into a local maximum.
    - Simulated-annealing-like effect.
  - Example: the six queens system
    - No local maxima (are proved to be) exist when *Nc* = 1.
    - Local maxima exist when *Nc* = 4 (global rule).



A local maximum of the six queens system

Locality control by rule composition*

- **The locality can also be controlled by composing rules.**
- **A rule with two or more catalysts may be composed using rules with one catalyst.**
  - Example: the *N* queens rule with two catalysts can be composed using the rule with one catalyst twice.



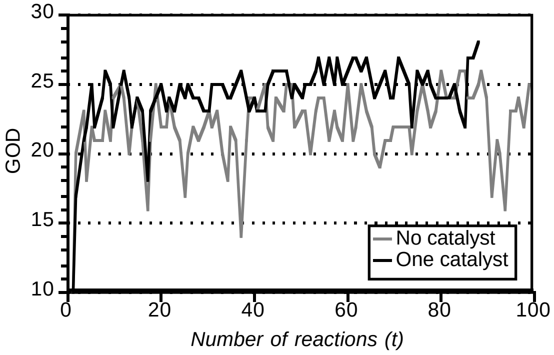
Swap Q2, Q3 by a rule with a catalyst — **the composed rule**

	State 0		State 1		State 2		State 3
Q1:	*c*1, *r*1	Swap Q1, Q2 → by a rule without a catalyst	**c2**, *r*1	Swap Q2, Q3 → by a rule without a catalyst	*c*2, *r*1	Swap Q1, Q3 → by a rule without a catalyst	**c1**, *r*1
Q2:	*c*2, *r*2		**c1**, *r*2		**c3**, *r*2		*c*3, *r*2
Q3:	*c*3, *r*3		*c*3, *r*3		**c1**, *r*3		**c2**, *r*3

Global order degree and its time sequence*

A macroscopic model of computation

- **Global order degree (GOD)**
  - GOD is the sum of the LODs of all the atoms (or all pairs of atoms).
  - The GOD is at a maximum at the solutions.
- **An example: the eight queens system**
  - 0 ≤ GOD ≤ 28.
  - The initial GOD is 0 — all the queens are on a diagonal line.



Other applications*

- **Current applications of CCM — still far from real world**

Classification		Problem	Rules and LODs		Performance	
			Number of rules*	Number of LODs	Time	Solution quality
NP-hard	Optimization	TSP	1	1	$O(N^3)$	97 times optimum out of 100 trials (N = 10)
		0–1 Knapsack	1 (or 2)	1	$O(N^2)$	45 times optimum out of 100 trials (N = 20)
	Constraint satisfaction	N Queens	1	1	$O(N^{4.6})$	–
		Graph (or map) coloring	1	1	–	–
P-hard		Sorting	1	1	$O(N^2)$	–

* Rules for working memory initialization are not counted.

- **The above problems are solved using very simple casters.**

Summary

- **I explained the self-oreganization paradigm.**
  - Self-organization — “global order” from computation with local information
- **We proposed a computation model CCM for self-organizing computation.**
  - Problems can be solved using one or a few simple production rules and evaluation functions.
  - Both production rules and evaluation functions works locally — i.e., on a small number of objects.
  - Locality of data reference can be controlled
    - By adding/removing catalysts and composing rules.
    - Local maxima can be avoided by changing locality.
    - Efficiency of searches can be controlled by changing locality.

Future work

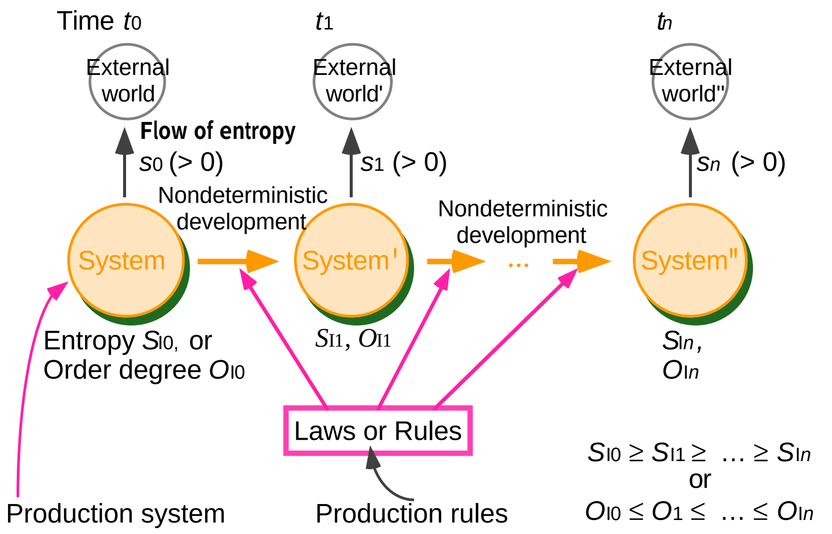
- **Toward open systems**
  - To develop CCM-based open systems
    - Constraint satisfaction or optimization problems are basically closed.
  - To observe and to analyze more complex emergent properties in those systems.
- **Self-referencial systems: a type of self-organizing systems**
  - To study self-modifying rules and LODs.
  - To study self-modifying targets of computation.
  - CCM must be enhanced to express self-references.

Contents*

- **Introduction — the self-organization paradigm**
- **Computation model CCM (Chemical Casting Model)**
- **Example: the N queens system**
- **Locality control of data references**
- **Other examples**
- **Summary and future work**

A model of self-organizing systems — 1*

A macroscopic model



Time t_0 t_1 t_n

External world

External world'

External world''

Flow of entropy

s_0 (> 0)

s_1 (> 0)

s_n (> 0)

Nondeterministic development

Nondeterministic development

System

System'

…

System''

Entropy S_{I0}, or Order degree O_{I0}

S_{I1}, O_{I1}

S_{In}, O_{In}

Laws or Rules

$S_{I0} \geq S_{I1} \geq \ldots \geq S_{In}$
or
$O_{I0} \leq O_{I1} \leq \ldots \leq O_{In}$

Production system

Production rules

A model of self-organizing systems — 2*

- **This model can be applied to a wide range of self-organizing systems, such as**
  - Our target self-organizing computational system.
  - A thermodynamic system that generates a dissipative structure.
- **The growth of a self-organizing system is autonomous, and, thus, its behavior is unpredictable, or it is observed as nondeterministic or driven by noise that comes from the outside of the system.**

Data in CCM*

Components of CCM — 3

- **Working memory**
  - The set of objects to which the rules apply.
- **Atoms**
  - Atoms are unit objects.
  - Atoms have internal state.
- **Links**
  - Links are connectors of atoms.
  - Links may have directions.
  - Links may have labels (names).

Order of reactions*

- **Order of reactions is nondeterministic.**
  - Random, or independent of the problem logic.
- **Different reaction orders may cause different results.**
  - All possible results will be as expected — because induced by the LODs.
- **Scheduling strategies**
  - Are specified by the user, or determined by the system.
  - Control the selections macroscopically.
  - Are similar to conflict resolution strategies in conventional production systems.

Types of scheduling strategies*

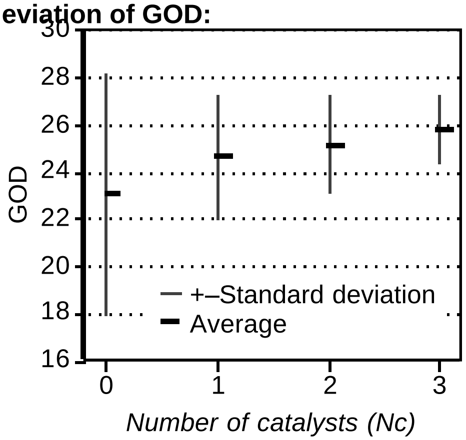
- **Mathematical random strategies (MRS)**
  - Use pseudo-random numbers.
  - Do not cause limit cycles, even if the user pays no attention.
  - Are the standard strategies.
- **Systematic strategies (SS)**
  - Use systematic methods — independent of the problem logic.
  - May cause limit cycles (infinite loops).
- **Parallel strategies**

Computation as Markov process*

- **Computation can be regarded as a stochastic process in CCM even when an S strategy is used.**
- **Three states during the computation of CCM.**
  - Strongly non-stationary state
    - The state in which the probability distribution rapidly changes when a reaction occurs.
  - Quasi-stationary state
    - The state that the probability of the solution state, $p(g_{max})$, increases when a reaction occurs, where gmax is the maximum value of the GOD (= NC2), but that the ratio of other states, $p(g)/(1 - p(g_{max}))$ ($g = g_{min}, \ldots, g_{max}-1$), are almost constant when a reaction occurs, where g_{min} is the minimum value of the GOD (= 0).
  - Termination state (Stationary state)
    - The state that $p(g_{max})$ is 1. This is the limit state when $t \to \infty$.
- **The above states can be modeled by a Markov chain.**

Effect of catalysts on GOD*

- **The average and standard deviation of GOD:**



- **When *Nc* increases,**
  - The average becomes higher.
  - The standard deviation becomes lower.
- **Catalysts bias the search.**
  - A rule with more catalysts searches among the states where the GOD is higher.
  - So the number of reactions is smaller.

Conflict and Cooperation in CCM*

Categories of CCM-based systems

- **Cooperative systems**
  - No reaction will decrease the GOD in cooperative systems.
  - Cooperative systems are called such because reactions cooperate toward the local or global maximum of the GOD.
  - Examples: TSP system, the 0–1 Knapsack system and the sorting systems.
- **Conflicting systems**
  - A reaction may decrease the GOD in conflicting systems.
  - Conflicting systems are called such because reactions does not cooperate toward that.
  - Some systems have little conflict while others have considerably more.
  - Examples: the *N* queens system and the graph coloring system.