# Problem Solving by General Purpose Solvers

Toshihide IBARAKI

Kwansei Gakuin University

# Topics in This Talk

1. General Purpose Solvers

2. Experience with Timetabling Competition

There are many different problems
in the real world

↓

Even if solvable by appropriate
OR approaches, we lack enough
man power and time

↓

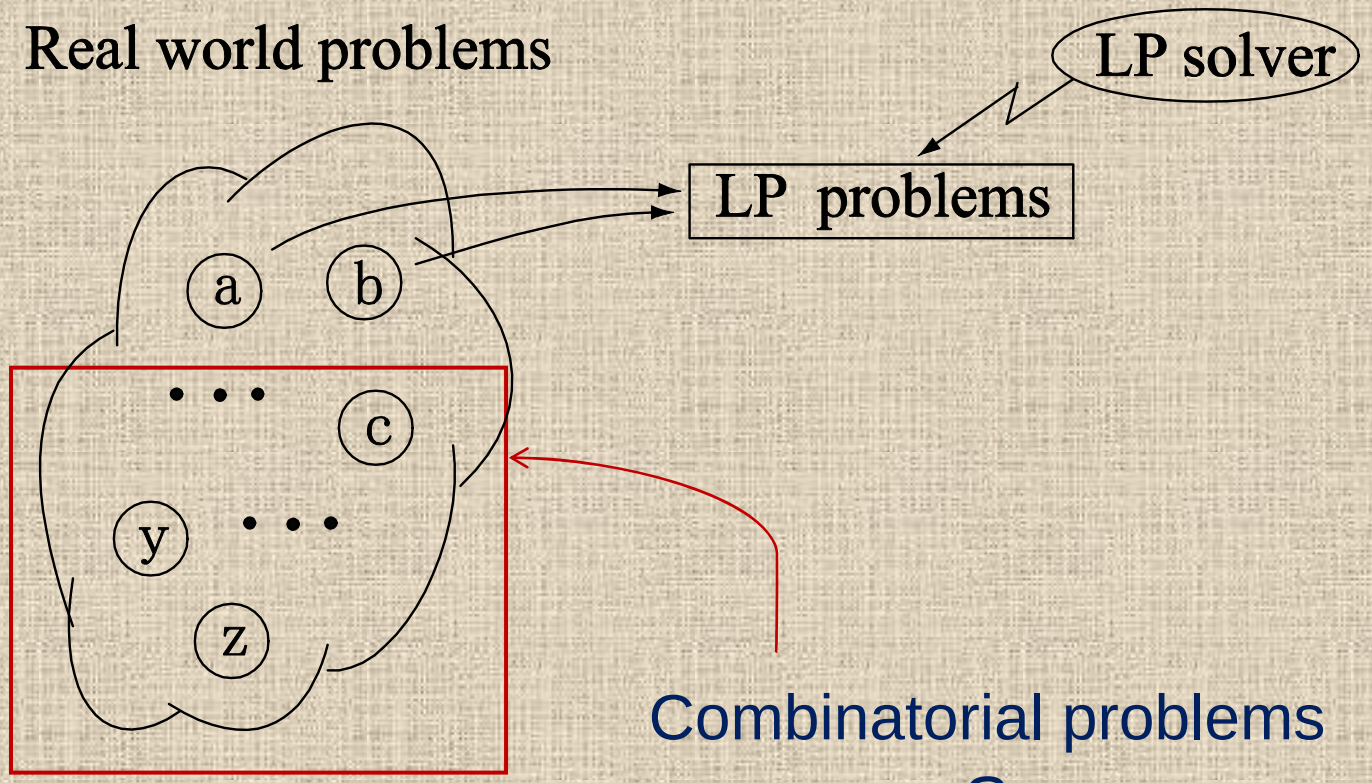
General purpose solvers
may save this situation

# A well known general solver

## Linear Programming (LP)

$$
\begin{aligned}
&Maximize && \sum_{j=1}^{n} c_j x_j \\
&subject\ \ to && \sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i = 1, 2, \ldots, m \\
& && x_j \geq 0, \qquad\quad j = 1, 2, \ldots, n
\end{aligned}
$$

# A first view of problem solving



# Combinatorial optimization problems

- Much wider application areas than LP
- Many problems in real world are NP-hard.
- NP hardness barrier:

  Under the hypothesis of$P \neq NP$ , NP hard problems cannot be solved in polynomial time.

# However,

- NP hardness is based on worst-case theory.

  Many problems may be solved in practical time.

  → E.g. Integer Programming (IP)

- Computing approximate solutions is not NP hard.

  Good approximate solutions are sufficient in practice.

  Approximate solutions may be obtained efficiently.

**Our recent view: NP hard problems can be solved efficiently for practical purposes.**

# IP problem

$$Maximize \quad \sum_{j=1}^{n} c_j x_j$$

$$subject \ \ to \quad \sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i = 1, 2, \ldots, m$$

$$x_j \geq 0, \qquad j = 1, 2, \ldots, n$$

$$x_j : \text{integers}, \quad j = 1, 2, \ldots, n$$

# Problem solving by IP

- Recent impressive progress of IP

  Branch-and-bound, branch-and-cut,

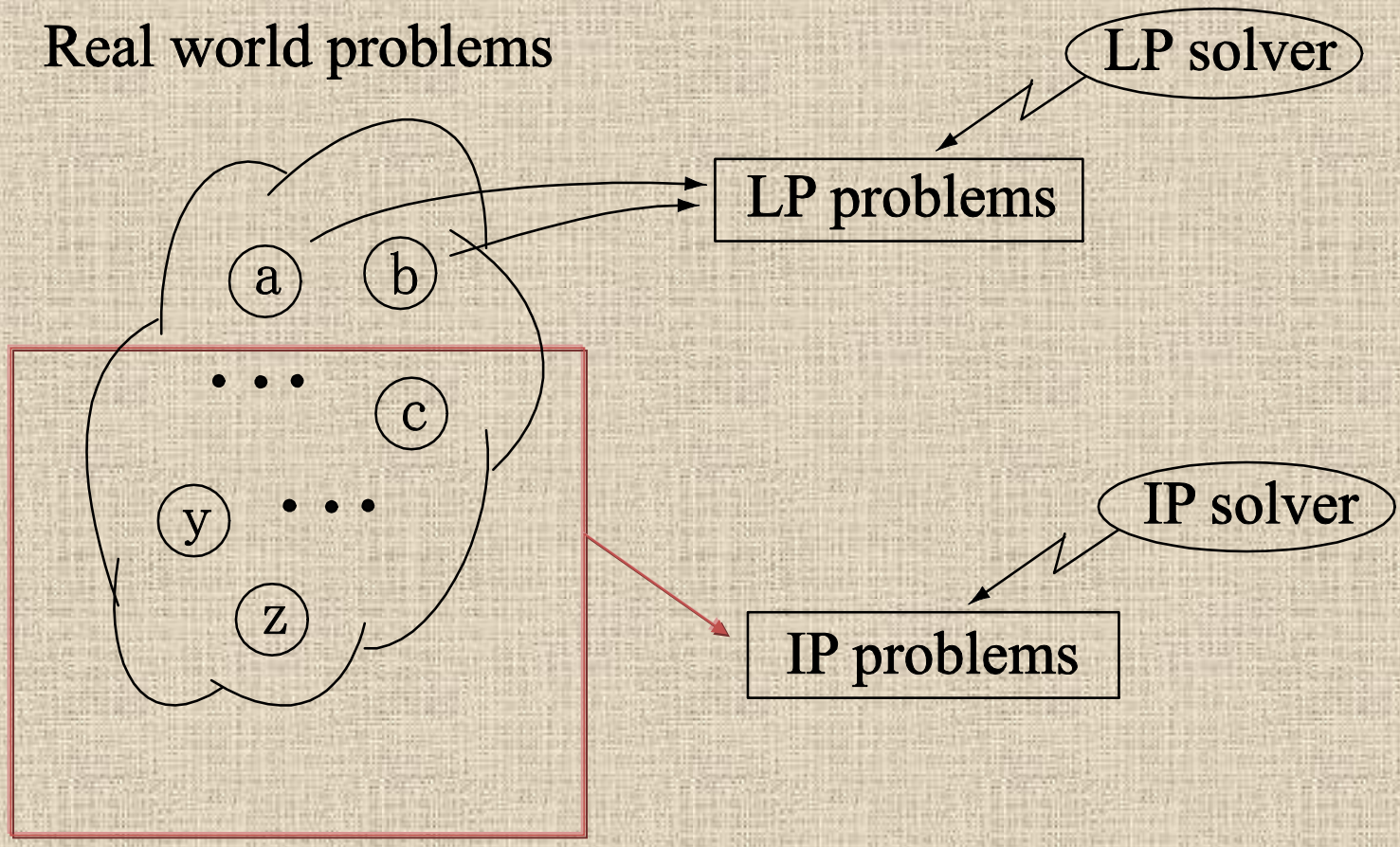
  cutting planes, integer polyhedra,

  commercial packages

- Theory of NP hardness tells that all problems in NP can be formulated as IP.

  MIP it!

# Second view of problem solving




Combinatorial optimization

# Still, however,

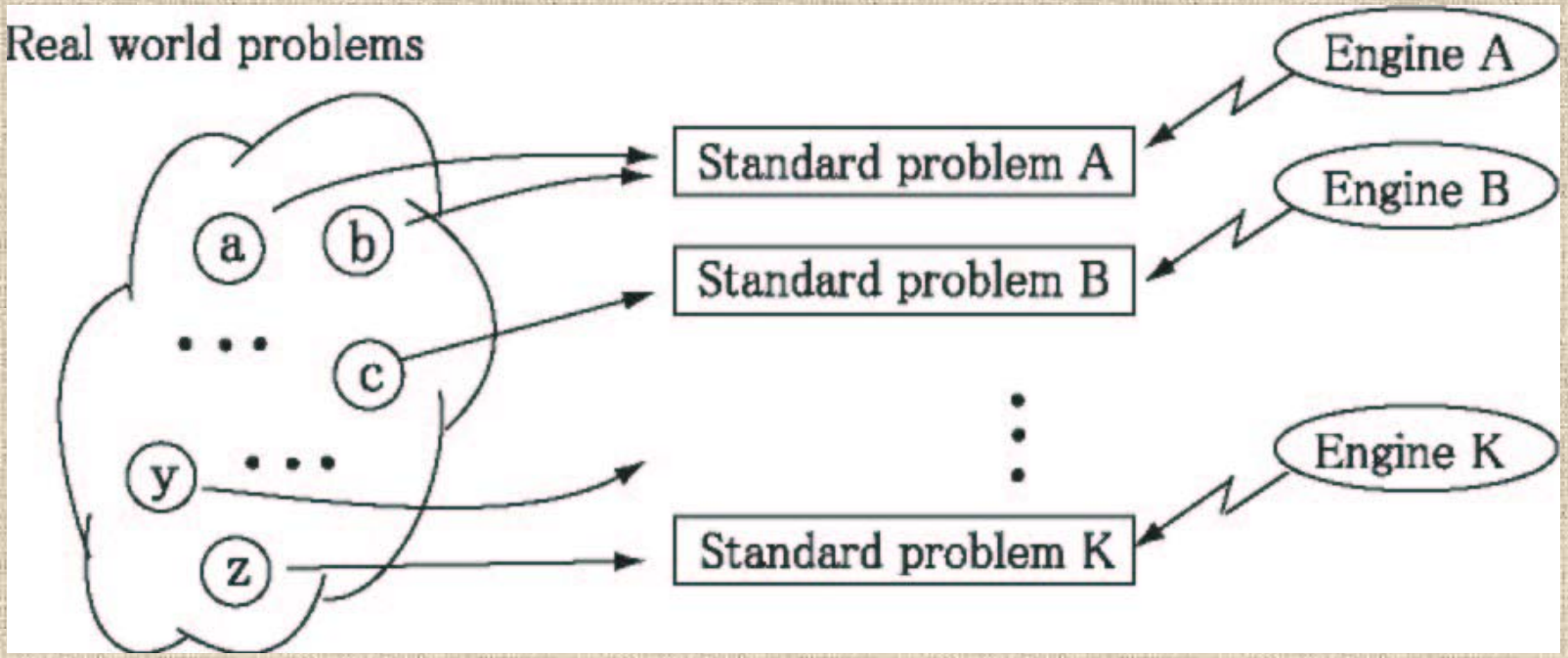
- Formulation as IP allows using additional variables and constrains of polynomial sizes

  Number of variables $n$ may become $n^2$ or $n^3$, etc.

  Similarly for the number of constraints.
- Usefulness of IP depends on problem types
- IP appears weak for problems with complicated combinatorial constraints and problems of scheduling type, for example.

# A last view of problem solving

- IP solver alone is not sufficient. Different types of solvers are needed.



# Standard Problems

- Should cover wide spectrum of problems

  Important problems in the real world.

- Should allow flexible formulations

  Various objective functions, additional constraints, soft constraints, . . .

- Should have structures that permit effective algorithms

  High efficiency, large scale problems, . . .

# List of Standard Problems

- Linear programming (LP)
- Integer programming (IP)
- Constraint satisfaction problem (CSP)
- Resource constrained project scheduling problem (RCPSP)
- Vehicle routing problem (VRP)
- 2-dimensional packing problem (2PP)
- Generalized assignment problem (GAP)
- Set covering problem (SCP)
- Maximum satisfiability problem (MAXSAT)

# Algorithms for general purpose solvers (approximate algorithms)

- Should have high efficiency, generality, robustness, flexibility, . . .

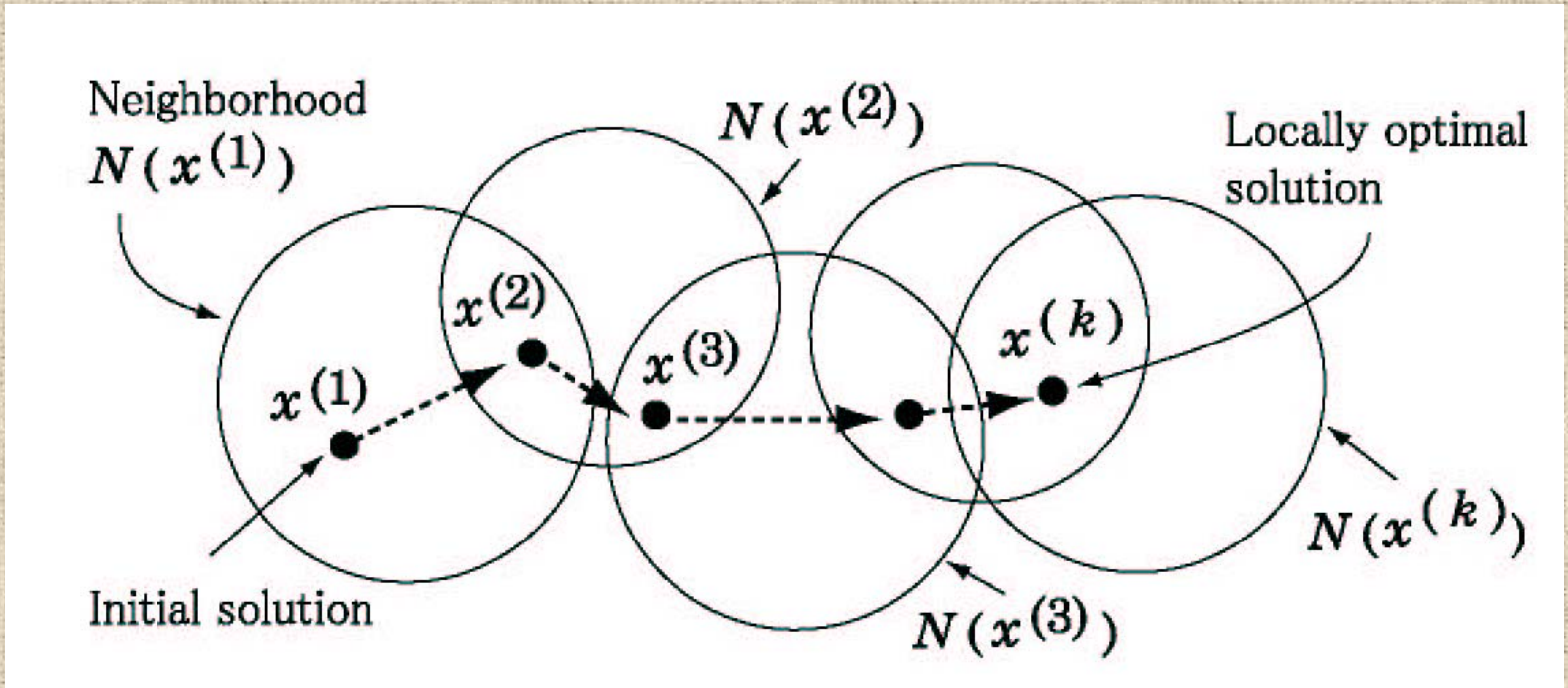
  Can such algorithms exist?

YES!

- Local search (LS)
- Metaheuristics

# Local search

- Starts from an appropriate initial solution.
- Repeats the operation of replacing the current solution by a better solution found in the neighborhood, as long as possible



# Framework of metaheuristics

**Step 1:** Generate an initial solution (based on the computational history so far).

**Step 2:** Apply (generalized) local search to find a good locally optimal solution.

**Step 3:** Halt if convergence condition is met, after outputting the best solution found so far. Otherwise return to Step 1.

Step 1 -- random generation, mutation, cross-over operation, path relinking, …, from a pool of good solutions obtained so far.

Step 2 -- simple local search, random moves with controlled probability, best moves with a tabu list, search with modified objective functions (e.g., with penalty of infeasibility), ...

# Typical metaheuristic algorithms

- Genetic algorithm
- Simulated annealing
- Tabu search
- Iterated local search
- Variable neighborhood search

. . .

- All of our solvers for standard problems have been constructed in the framework of metaheuristics, in particular tabu search.

# Experience with Timetabling

# ITC **2007**

International timetabling competition sponsored by PATAT and WATT (second competition)

- **Track 1:** Examination timetabling
- **Track 2:** Post enrolment based course timetabling
- **Track 3:** Curriculum based course timetabling

It is required to obtain solutions that satisfy all hard constraints; competition is made to minimize the penalties of soft constraints.

# Procedure of ITC2007

1. Benchmark problems in three tracks are made public.
2. Participants solve benchmarks on their machines, using the time limit specified by the code provided by the organizers, and submit their results.
3. Organizers select five finalists in each track.
4. Finalists send their executable codes to the organizers, who then test the codes on a set of hidden benchmarks.
5. Organizers announce finalists orderings.
6. Winners are invited to PATAT2008.

# Track 1: Examination timetabling

- Input data: Set of examinations, set of rooms, set of periods, set of registered students for each exam, where exams and periods have individual lengths.
- Assignment of all exams to rooms and periods is asked.
- Rooms have capacities, and more than one exam can be assigned to a room.
- All students can take all registered exams.
- Desirable to avoid consecutive exams and to space $\alpha$ periods between two successive exams, for each student.
- Exams assigned to a room are better to have the same length.
- Problem sizes: 200-1000 exams, 5000-16000 students, 20-80 periods, and 1-50 rooms.

# Track 2: Post enrolment based course timetabling

- Input data: Set of lectures, set of rooms, 45 periods (5 days x 9 periods), set of registered students for each lecture.
- Rooms have capacities and features, and at most one lecture is assigned to a room which satisfies capacity and has required features.
- Lectures not to be assigned to the same period are specified.
- All students can take all registered lectures.
- Desirable to avoid the last period of each day.
- Desirable to avoid three consecutive lectures for each student.
- Desirable to avoid one lecture a day for each student.
- Problem sizes: 200-400 lectures, 300-1000 students, 10-20 rooms.

# Track 3: Curriculum based course timetabling

- Input data: Set of curriculums, set of rooms, set of periods and the number of students in each curriculum. Each curriculum contains a set of courses, and each course contains a set of lectures.
- Rooms have capacities, and at most one lecture is assigned to a room.
- Desirable to distribute lectures of one course evenly in a week.
- Desirable to congregate the lectures in a curriculum each day.
- Problem sizes: 150-450 lectures, 25-45 periods, 5-20 rooms.

# Formulation as CSP

- CSP uses variables $X_i$ with domain $D_i$, and value variables $x_{ij}$ (taking 1 if $X_i=j \in D_i$ and 0 otherwise).
- CSP allows any constraints, particularly linear and quadratic inequalities and equalities using value variables, and all_different constraints of variables.
- Our CSP solver is based on tabu search.

# Notations

- Indexes: $i$ for lectures, $j$ for periods, $l$ for students and $k$ for rooms.
- $X_i$ has domain $P_i$ (set of possible periods of $i$),
  $Y_i$ has domain $R_i$ (set of possible rooms of $i$).
  $x_{ij} = 1(0)$ if $i$ is (not) assigned to period $j$,
  $y_{ik} = 1(0)$ if $i$ is (not) assigned to room $k$.

# Hard constraints

- Capacity constraints of rooms:

$$\sum_i s_i x_{ij} y_{ik} \le r_k, \quad \forall j,k$$

$$\sum_i x_{ij} y_{ik} \le 1, \quad \forall j,k$$

- If a student $l$ takes lectures $i_1, i_2, \ldots, i_a$:

$$\text{All\_different}\ (X_{i1}, X_{i2}, \ldots, X_{ia})$$

- Variable $X_i$ enforces that $i$ is assigned to exactly one period.
- Similarly for other hard constraints.

# Soft constraints

- A student $l$ does not take exams in two consecutive periods: $\sum_{i \in E_l} (x_{ij} + x_{i(j+1)}) \leq 1, \quad \forall l, \forall j$
- The number of lectures for student $l$ is either 0 or more than 1:

$$\sum_{i \in E_l} \sum_{j_h} x_{i((j_d - 1)h + j_h)} \leq h \cdot z_{j_d l}, \quad \forall j_d, l$$

$$\sum_{i \in E_l} \sum_{j_h} x_{i((j_d - 1)h + j_h)} \geq 2 z_{j_d l}, \quad \forall j_d, l$$

- Similarly for other constraints.

# Formulation as IP

- All hard and soft constraints can be written as linear inequalities or equalities, if additional variables and constraints are introduced.
- The number of such additions are enormous since they correspond to nonlinear terms in CSP formulations.
- IP is not appropriate for these timetabling problems of large sizes, because of their complicated constraints.

# ITC2007 Results

international TIMETABLING COMPETITION

sponsored by PATAT



REGISTER NOW
Already registered? Click here to login


## Contact Details

Dr Barry McCollum
SARC Building
School of Electronics, Electrical Engineering & Computer Science
Queen's University Belfast

Phone: +44 (0) 2890974622
Fax: +44 (0) 2890975666
Email: b.mccollum@qub.ac.uk


## Finalist Ordering

The following information details the finalists for each track **in place order.**

Please note that a report detailing the background to the competition can be found here. This has been submitted for consideration to INFORMS Journal on Computing.

### Examination Track

Best recorded scores may be viewed here. By clicking on individual names more details relating to scores are available.

1st Place: Tomas Müller (USA)

2nd Place: Christos Gogos (Greece)

3rd Place:Mitsunori Atsuta, Koji Nonobe, and Toshihide Ibaraki (Japan)

4th Place: Geoffrey De Smet (Belgium)

5th Place: Nelishia Pillay (South Africa)

### Post Enrolment based Course Timetabling

An excell spreadsheet containing all the scores can be downloaded here. This information is also available as .csv or.xml format.

1st Place: Hadrien Cambazard, Emmanuel Hebrard, Barry O'Sullivan, Alexandre Papadopoulos (Ireland)

2nd Place: Mitsunori Atsuta, Koji Nonobe, and Toshihide Ibaraki (Japan)

3rd Place: Marco Chiarandini, Chris Fawcett, Holger H Hoos (Denmark)

4th Place: Clemens Nothegger, Alfred Mayer, Andreas Chwatal, Gunther Raidl (Austria)

### Curriculum based Course Timetabling

An excell document containing all the scores can be found here. This information is also available as .csv or.xml format.

1st Place: Tomas Müller (USA)

2nd Place: Zhipeng Lu and Jin-Kao Hao (France)

3rd Place: Mitsunori Atsuta, Koji Nonobe, and Toshihide Ibaraki (Japan)

4th Place: Martin Josef Geiger (Germany)

5th Place: Michael Clark, Martin Henz, and Bruce Love (Singapore)

# Conclusion and discussion

- Our experience with ITC2007 tells that the general purpose solvers can handle wide types of problems in the practical sense.
- Other applications: Industrial applications, Academic applications
- Commercial package NUOPT (Mathematical Systems, Inc.)

# Acknowledgment


- Challenge to ITC2007 was conducted by M. Atsuta (NS Solutions Corp.), Koji Nonobe (Hosei University) and T.I.
- CSP solver was developed by Koji Nonobe and T.I.

K. Nonobe and T. Ibaraki, An improved tabu search method for the weighted constraint satisfaction problem, INFOR, 39, pp. 131-151, 2001.

Thank you for your attention

# Theory of NP hardness

- Class NP contains almost all combinatorial problems of practical interest.
- NP-hard problems are most difficult ones in NP.
- Most of problems we encounter are NP hard.
- If one of NP-hard problems can be solved in polynomial time, then all problems in NP are solvable in polynomial time, which is most unlikely.

→ $P \neq NP$ conjecture

# Ingredients of local search (LS)

- Solution space and search space
- Neighborhood

  High possibility of containing improved solutions
- Reduction of neighborhood size

  Removing unnecessary solutions in advance
- Search method in the neighborhood

  Random, fixed ?

  Best improvement, first improvement ?