Challenging combinatorial problems

Yves Crama

HEC Management School
University of Liège

Francqui Lecture, KUL, May 2010

Outline

Objectives:

Outline

Objectives:

- recall some challenging combinatorial problems from previous lectures

Outline

Objectives:

- recall some challenging combinatorial problems from previous lectures
- introduce a couple of new ones

Outline

Objectives:

- recall some challenging combinatorial problems from previous lectures
- introduce a couple of new ones
- difficulty: worth a variable number of bottles of wine (scale: from 1 to ∞).

Outline

1. Combinatorial models in manufacturing

Outline

Outline

1. Combinatorial models in manufacturing
   - Tool management
   - Robotic cells
2. Boolean functions and games
   - Dualization
   - Threshold functions and weighted majority games

Outline

1. Combinatorial models in manufacturing
   - Tool management
   - Robotic cells
2. Boolean functions and games
   - Dualization
   - Threshold functions and weighted majority games

Setup minimization

Tool setup in automated manufacturing:

- limited size of the tool magazine (say, 10 to 120 tools)

Setup minimization

Tool setup in automated manufacturing:

- limited size of the tool magazine (say, 10 to 120 tools)
- many more tools may be stored in a central storage area

Setup minimization

Tool setup in automated manufacturing:

- limited size of the tool magazine (say, 10 to 120 tools)
- many more tools may be stored in a central storage area
- transferred to the machines as required

Setup minimization

Tool setup in automated manufacturing:

- limited size of the tool magazine (say, 10 to 120 tools)
- many more tools may be stored in a central storage area
- transferred to the machines as required
- costly, slow, error-prone operations

Setup minimization

Tool setup in automated manufacturing:

- limited size of the tool magazine (say, 10 to 120 tools)
- many more tools may be stored in a central storage area
- transferred to the machines as required
- costly, slow, error-prone operations

One-machine scheduling with tooling decisions:

Simultaneously

- sequence parts to be processed and
- allocate tools required to the machine

so as to minimize tool setup costs.

Setup minimization

Various models for setup minimization.

Common data

- M: number of tools;
- N : number of parts;
- A : $M \times N$ tool-part matrix:

  $$a_{ij} = 1 \text{ if part } j \text{ requires tool } i, 0 \text{ otherwise;}$$
- C : capacity of the tool magazine (= number of tool slots)

Setup minimization

Various models for setup minimization.

Common data

- M: number of tools;
- N : number of parts;
- A : $M \times N$ tool-part matrix:

$$a_{ij} = 1 \text{ if part } j \text{ requires tool } i\text{, 0 otherwise;}$$

- C : capacity of the tool magazine (= number of tool slots)

Feasible batch

A batch of parts is feasible if it can be carried out without tool switches, i.e., if it requires at most C tools.

Example

Capacity: $C = 3$

Parts

Tools

	P_1	P_2	P_3	P_4	P_5
T_1	1	0	1	0	1
T_2	1	0	0	1	0
T_3	0	1	1	1	0
T_4	0	1	0	0	1

Example

Capacity: $C = 3$

Tools / **Parts**

	P_1	P_2	P_3	P_4	P_5
T_1	1	0	1	0	1
T_2	1	0	0	1	0
T_3	0	1	1	1	0
T_4	0	1	0	0	1

Batch selection model

Batch selection

Find a feasible batch of maximum cardinality.

Batch selection model

Batch selection

Find a feasible batch of maximum cardinality.

Equivalently:

Batch selection

Find a largest subset of columns of the tool-part incidence matrix such that the submatrix induced by this subset has at most C nonzero rows.

Batch selection model

Batch selection

Find a feasible batch of maximum cardinality.

Equivalently:

Batch selection

Find a largest subset of columns of the tool-part incidence matrix such that the submatrix induced by this subset has at most C nonzero rows.

or...

Batch selection

Given a hypergraph, find a subset of C vertices that contains the largest possible number of hyperedges.

Pseudo-Boolean formulation

- define : $x_i = 1$ if tool i is selected, $x_i = 0$ otherwise

Pseudo-Boolean formulation

- define : $x_i = 1$ if tool i is selected, $x_i = 0$ otherwise
- part j can be processed if and only if $x_i = 1$ for all tools i such that $a_{ij} = 1$

Pseudo-Boolean formulation

- define : $x_i = 1$ if tool i is selected, $x_i = 0$ otherwise
- part j can be processed if and only if $x_i = 1$ for all tools i such that $a_{ij} = 1$
- equivalently: part j can be processed if and only if $\prod_{i:a_{ij}=1} x_i = 1$

Pseudo-Boolean formulation

- define : $x_i = 1$ if tool i is selected, $x_i = 0$ otherwise
- part j can be processed if and only if $x_i = 1$ for all tools i such that $a_{ij} = 1$
- equivalently: part j can be processed if and only if $\prod_{i:a_{ij}=1} x_i = 1$

So:

Nonlinear knapsack

Batch selection is equivalent to the *nonlinear (supermodular) knapsack problem*

$$\max \quad \sum_{j=1}^{N} \prod_{i:a_{ij}=1} x_i$$
$$\text{subject to } \sum_{i=1}^{M} x_i \leq C, \quad (x_1, ..., x_M) \in \{0,1\}^M$$

Complexity

Theorem

Batch selection is NP-hard.

Generalization of maximum clique.

Complexity

Theorem

Batch selection is NP-hard.

Generalization of maximum clique.

Many papers on this problem:

- integer programming
- heuristics
- special “graphical” case
- subproblem for *part grouping problem*: partition parts into small number of feasible batches (minimize setups).

Worst-case ratio

From a theoretical point of view, one may ask:

Worst-case ratio

What is the (theoretical) worst-case ratio of heuristics for the batch selection problem, where:

$$wcr = \frac{optimal\ value}{heuristic\ value}?$$

Analysis

Crama and van de Klundert (1999) proved:

Theorem

If there is a polynomial-time approximation algorithm with constant worst-case ratio for batch selection, then there is also a polynomial-time approximation scheme for this problem.

... meaning roughly that the optimal value can be approximated arbitrarily closely in polynomial time.

Conjectures

Conjecture

There exists no polynomial-time approximation algorithm with constant worst-case ratio for batch selection, unless P = NP.

Conjectures

Conjecture

There exists no polynomial-time approximation algorithm with constant worst-case ratio for batch selection, unless P = NP.

Conjectures

Perhaps even true:

Conjecture

There exists no polynomial-time approximation algorithm with worst-case ratio $O(poly(C))$ for batch selection, unless P = NP.

Conjectures

Perhaps even true:

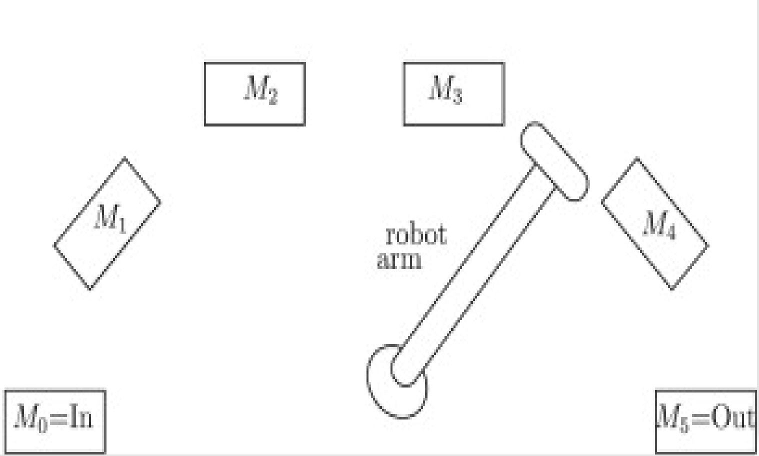
Conjecture

There exists no polynomial-time approximation algorithm with worst-case ratio $O(poly(C))$ for batch selection, unless P = NP.

Outline

1. Combinatorial models in manufacturing
   - Tool management
   - Robotic cells
2. Boolean functions and games
   - Dualization
   - Threshold functions and weighted majority games

Circular robotic cell



Robotic cell flowshops

- m machines in line (or on a circle), without buffer space: $M_1, M_2, \ldots, M_m$
- loading station M_0 and unloading station M_{m+1}
- set of parts to be produced by the line
- a unique robot loads and unloads the parts

Robotic cell flowshops

- m machines in line (or on a circle), without buffer space: $M_1, M_2, \ldots, M_m$
- loading station M_0 and unloading station M_{m+1}
- set of parts to be produced by the line
- a unique robot loads and unloads the parts

Robotic cell scheduling

Determine

- a sequence of parts,
- a robot activity sequence,
- a production schedule (start/end times),
- so as to minimize cycle time (maximize throughput).

Assumptions

We concentrate here on:

- repetitive production of identical parts
- no intermediate buffers
- additive robot travel times.

1-Unit cycles

1-Unit cycles

A 1-unit cycle is a sequence of activities which unloads exactly one part in the output buffer and which returns the cell to its initial state.

(In particular, every activity is performed exactly once and the cycle can be repeated indefinitely.)

1-Unit cycles

1-Unit cycles

A 1-unit cycle is a sequence of activities which unloads exactly one part in the output buffer and which returns the cell to its initial state.

(In particular, every activity is performed exactly once and the cycle can be repeated indefinitely.)

Crama and van de Klundert (1997) proved:

Theorem

For a robotic cell with m machines, a 1-unit cycle that minimizes the average cycle time can be computed in $O(m^3)$ time.

Optimality of 1-unit cycles

But... 1-unit cycles do not necessarily minimize the cycle time.

Optimality of 1-unit cycles

But... 1-unit cycles do not necessarily minimize the cycle time.

k-Unit cycles

A k-unit cycle is a sequence of activities which unloads exactly k parts in the output buffer and which returns the cell to its initial state.

Optimality of 1-unit cycles

But... 1-unit cycles do not necessarily minimize the cycle time.

k-Unit cycles

A *k-unit cycle* is a sequence of activities which unloads exactly k parts in the output buffer and which returns the cell to its initial state.

Note: For some k, some k-unit cycle is optimal.

Open question

Main open question:

Complexity

What is the complexity of computing an optimal (cyclic) robot move sequence?

Open question

Main open question:

Complexity

What is the complexity of computing an optimal (cyclic) robot move sequence?

Conjecture

Computing an optimal (cyclic) robot move sequence is NP-hard.

Open question

Main open question:

Complexity

What is the complexity of computing an optimal (cyclic) robot move sequence?

Conjecture

Computing an optimal (cyclic) robot move sequence is NP-hard.

Outline

1. Combinatorial models in manufacturing
   - Tool management
   - Robotic cells

2. Boolean functions and games
   - Dualization
   - Threshold functions and weighted majority games

Outline

1. Combinatorial models in manufacturing
   - Tool management
   - Robotic cells
2. Boolean functions and games
   - Dualization
   - Threshold functions and weighted majority games

Definitions

Recall

Boolean functions

A Boolean function is a mapping $\varphi : \{0,1\}^n \rightarrow \{0,1\}$. Function φ is *positive* (monotone, isotone) if

$$X \leq Y \Rightarrow \varphi(X) \leq \varphi(Y).$$

Definitions

Recall

Boolean functions

A Boolean function is a mapping $\varphi : \{0,1\}^n \rightarrow \{0,1\}$. Function φ is *positive* (monotone, isotone) if

$$X \leq Y \Rightarrow \varphi(X) \leq \varphi(Y).$$

Set functions:

Boolean functions on $\{0,1\}^n$ can also be viewed as *set functions*, that is, functions defined on subsets of $\{1,2,\ldots,n\}$.

Example

x_1	x_2	x_3	S	φ
0	0	0	$\emptyset$	0
0	0	1	$\{3\}$	1
0	1	0	$\{2\}$	0
0	1	1	$\{2,3\}$	1
1	0	0	$\{1\}$	0
1	0	1	$\{1,3\}$	1
1	1	0	$\{1,2\}$	1
1	1	1	$\{1,2,3\}$	1

Simple games

- A positive Boolean function φ defines a *simple game* or *voting game*.
- Interpretation: φ describes the voting rule which is adopted by the players when a decision is to be made.
- If S is a subset of players, then $\varphi(S)$ is the outcome of the voting process when all players in S say "Yes".
- Positivity means:

$$S \subseteq T \Rightarrow \varphi(S) \leq \varphi(T).$$

Mimimal true points

- A positive Boolean function φ, or a simple game, is completely defined by the list of its *mimimal true points*, that is, minimal subsets of players $S_1, S_2, ..., S_m$ such that $\varphi(S_i) = 1$ for $i = 1, 2, ..., m$.

Example

x_1	x_2	x_3	S	φ	MTP
0	0	0	$\emptyset$	0	
0	0	1	$\{3\}$	1	S_1
0	1	0	$\{2\}$	0	
0	1	1	$\{2,3\}$	1	
1	0	0	$\{1\}$	0	
1	0	1	$\{1,3\}$	1	
1	1	0	$\{1,2\}$	1	S_2
1	1	1	$\{1,2,3\}$	1	

Maximal false points

- A positive Boolean function φ, or a simple game, is completely defined by the list of its *mimimal true points*, that is, minimal subsets of players $S_1, S_2, ..., S_m$ such that $\varphi(S_i) = 1$ for $i = 1, 2, ..., m$.
- Similarly, φ is completely defined by the list of its *maximal false points*, that is, maximal subsets of players $T_1, T_2, ..., T_k$ such that $\varphi(T_j) = 0$ for $j = 1, 2, ..., k$.

Example

x_1	x_2	x_3	S	φ	MTP	MFP
0	0	0	$\emptyset$	0		
0	0	1	$\{3\}$	1	S_1	
0	1	0	$\{2\}$	0		T_1
0	1	1	$\{2,3\}$	1		
1	0	0	$\{1\}$	0		T_2
1	0	1	$\{1,3\}$	1		
1	1	0	$\{1,2\}$	1	S_2	
1	1	1	$\{1,2,3\}$	1		

Dualization

A fundamental algorithmic problem:

Dualization

- Input: the list of minimal true points of a positive Boolean function.
- Output: the list of maximal false points of the function.

(or conversely)

Problem investigated in Boolean theory, game theory, integer programming, electrical engineering, artificial intelligence, reliability, combinatorics, etc.

Dualization

Dualization amounts to generating

Dualization

Dualization amounts to generating

- all minimal transversals of a hypergraph (V, E), $E = (E_1, \ldots, E_m)$, $E_i \subseteq V$ (in particular: all maximal stable sets of a graph);

Dualization

Dualization amounts to generating

- all minimal transversals of a hypergraph (V, E), $E = (E_1, \ldots, E_m)$, $E_i \subseteq V$ (in particular: all maximal stable sets of a graph);
- all minimal solutions of a set covering problem:

$$\sum_{j \in E_i} x_j \geq 1 \ (i = 1, ..., m);$$

$$x_j \in \{0, 1\} \ (j = 1, ..., n).$$

Complexity

Note: the output is uniquely defined, but its size can be exponentially large in the size of the input.

Complexity

Note: the output is uniquely defined, but its size can be exponentially large in the size of the input.

(Lawler, Lenstra, Rinnooy Kan 1980; Johnson, Papadimitriou, Yannakakis 1988; Bioch, Ibaraki 1995; etc.)

Can positive Boolean functions be dualized in *total polynomial time*, that is, in time polynomial in the combined size of the input and of the output?

Equivalent problem

Dualization is “polynomially equivalent” to the problem:

Test Dual

- Input: the MTPs of a Boolean function φ, and a list L of points.
- Question: is L the list of MFPs of φ ?

Equivalent problem

Dualization is “polynomially equivalent” to the problem:

Test Dual

- Input: the MTPs of a Boolean function φ, and a list L of points.
- Question: is L the list of MFPs of φ ?

- Dualization can be solved in total polynomial time if and only if Test Dual can be solved in polynomial time.
- Test Dual does not require exponential outputs.

Quasi-polynomial algorithm

Fredman and Khachiyan have shown

Fredman and Khachiyan (1996)

Dualization can be solved in time $O(m^{\log m})$, where m is the combined size of the input and of the output of the problem.

Quasi-polynomial algorithm

Fredman and Khachiyan have shown

Fredman and Khachiyan (1996)

Dualization can be solved in time $O(m^{\log m})$, where m is the combined size of the input and of the output of the problem.

- Several generalizations of this result have been obtained by Boros, Elbassioni, Gurvich, Khachiyan, Makino, etc.
- But the central questions remain open:

Open problems

Complexity of Dualization

Can dualization be solved in total polynomial time?

Open problems

Complexity of Dualization

Can dualization be solved in total polynomial time?

Complexity of Test Dual

Can Test Dual be solved in polynomial time? Is it NP-hard (unlikely)?

Open problems

Complexity of Dualization

Can dualization be solved in total polynomial time?

Complexity of Test Dual

Can Test Dual be solved in polynomial time? Is it NP-hard (unlikely)?

Outline

1. Combinatorial models in manufacturing
   - Tool management
   - Robotic cells
2. Boolean functions and games
   - Dualization
   - Threshold functions and weighted majority games

Weighted majority games

Weighted majority games

A simple game φ is a *weighted majority game* if

- each player i carries a voting weight w_i

Weighted majority games

Weighted majority games

A simple game φ is a *weighted majority game* if

- each player i carries a voting weight w_i
- there is a voting threshold q

Weighted majority games

Weighted majority games

A simple game φ is a *weighted majority game* if

- each player i carries a voting weight w_i
- there is a voting threshold q
- $\varphi(S) = 1 \iff \sum_{i \in S} w_i > q$

Weighted majority games

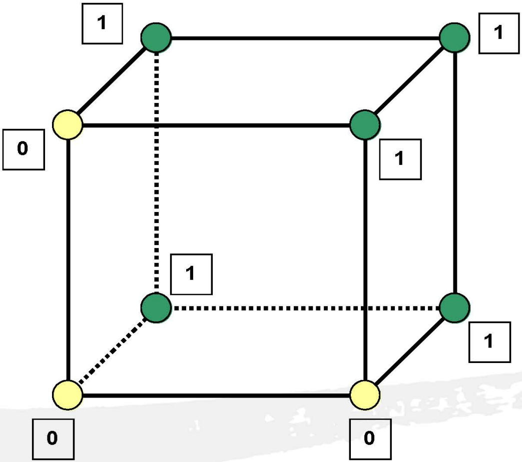
Weighted majority games

A simple game φ is a *weighted majority game* if

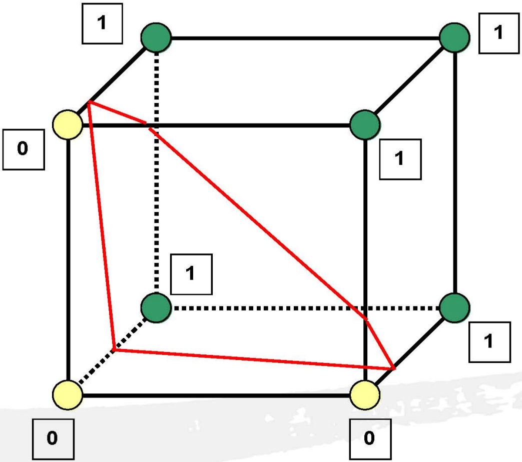
- each player i carries a voting weight w_i
- there is a voting threshold q
- $\varphi(S) = 1 \iff \sum_{i \in S} w_i > q$

In Boolean theory, a weighted majority game is called a *threshold function*.

Example



Example



Applications

Numerous applications:

- game theory (weighted voting)
- electrical engineering (gates)
- optimization (knapsack)
- neural networks (perceptrons)
- databases (concurrent access)
- etc.

Threshold recognition

A fundamental algorithmic problem:

Threshold recognition

- Input: the list of minimal true points of a positive Boolean function φ.
- Question: is φ a threshold function (weighted majority game)?

Threshold recognition

A fundamental algorithmic problem:

Threshold recognition

- Input: the list of minimal true points of a positive Boolean function φ.
- Question: is φ a threshold function (weighted majority game)?

Peled and Simeone (1985)

Threshold recognition can be solved in polynomial time.

Generic approach

Observations:

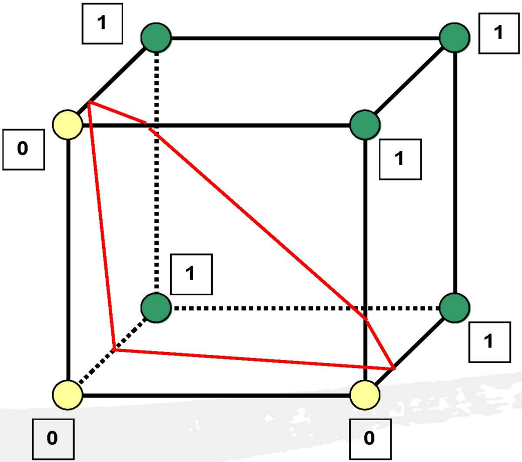
- every threshold function φ is *regular*: up to a permutation of its variables, $\varphi(\ldots 0 \ldots 1 \ldots) \leq \varphi(\ldots 1 \ldots 0 \ldots)$

Generic approach

Observations:

- every threshold function φ is *regular*: up to a permutation of its variables, $\varphi(\ldots 0 \ldots 1 \ldots) \leq \varphi(\ldots 1 \ldots 0 \ldots)$
- regular functions can be recognized and dualized in polynomial time (Peled-Simeone 1985, Crama 1989, etc.)
- recall: dualizing amounts to generating all maximal 0s from all minimal 1s.

Example



Generic approach

Recognition procedure:

Generic approach

Recognition procedure:

- test whether φ is *regular*

Generic approach

Recognition procedure:

- test whether φ is *regular*
- if so, dualize φ

Generic approach

Recognition procedure:

- test whether φ is *regular*
- if so, dualize φ
- solve the linear programming problem (in $w_1, \ldots, w_n$):

$$w_1 x_1 + w_2 x_2 + \ldots + w_n x_n \leq q \quad \text{if } \varphi(x_1, \ldots, x_n) = 0$$

$$w_1 x_1 + w_2 x_2 + \ldots + w_n x_n > q \quad \text{if } \varphi(x_1, \ldots, x_n) = 1$$

Generic approach

Recognition procedure:

- test whether φ is *regular*
- if so, dualize φ
- solve the linear programming problem (in $w_1, \ldots, w_n$):

$$w_1 x_1 + w_2 x_2 + \ldots + w_n x_n \leq q \quad \text{if } \varphi(x_1, \ldots, x_n) = 0$$

$$w_1 x_1 + w_2 x_2 + \ldots + w_n x_n > q \quad \text{if } \varphi(x_1, \ldots, x_n) = 1$$

Complexity

- $O(n^2 m)$ to test regularity and to dualize
- complexity of linear programming for separation

Open problem

Purely combinatorial recognition procedure

Can we avoid to solve an LP in order to recognize threshold functions?

Open problem

Purely combinatorial recognition procedure

Can we avoid to solve an LP in order to recognize threshold functions?

- Some failed attempts in that direction in the 60's (Chow, Winder, Dertouzos, etc.)
- Question could be taken up again in the light of advances in complexity theory and optimization.
- Recent attempts by Smaus (incomplete?)

Open problem

Purely combinatorial recognition procedure

Can we avoid to solve an LP in order to recognize threshold functions?

Conclusion

Combinatorial Models and Complexity in Management Science

- Many nice, challenging problems at the interface of applications and combinatorial mathematics.

Conclusion

Combinatorial Models and Complexity in Management Science

- Many nice, challenging problems at the interface of applications and combinatorial mathematics.
- Many more to be found in
  Y. Crama and P.L. Hammer, *Boolean Functions: Theory, Algorithms, and Applications*, Cambridge University Press, New York, to appear.
  Y. Crama and P.L. Hammer, eds., *Boolean Models and Methods in Mathematics, Computer Science and Engineering*, Cambridge University Press, New York, 2010.

Conclusion

Combinatorial Models and Complexity in Management Science

- Many nice, challenging problems at the interface of applications and combinatorial mathematics.
- Many more to be found in
  Y. Crama and P.L. Hammer, *Boolean Functions: Theory, Algorithms, and Applications*, Cambridge University Press, New York, to appear.
  Y. Crama and P.L. Hammer, eds., *Boolean Models and Methods in Mathematics, Computer Science and Engineering*, Cambridge University Press, New York, 2010.

Conclusion

Combinatorial Models and Complexity in Management Science

Thank you for your presence and for your attention!!