Flexible Optimization: Nurse Scheduling with Constraint Programming and Automata

Sophie Demassey

Centre de Mathématiques Appliquées, MINES ParisTech
http://sofdem.github.io/

CMP, Gardanne, 3 July 2014

mutability of practical recurring problems

example 1: online data center resource management

`http://btrp.inria.fr/` [Hermenier09]

mutability of practical recurring problems

example 2: employee timetabling

`https://github.com/sofdem/chocoETP` [Menana09]

outline

1. Mutable Problem
   - Nurse Scheduling
2. Flexible Tools
   - finite automata
   - global constraints
3. Flexible Solutions
   - multicost-regular = automata + global constraints
   - ChocoETP = automata + CP + local search
4. Conclusion

Nurse Scheduling Problem

an illustration of mutability

Nurse Scheduling Problem

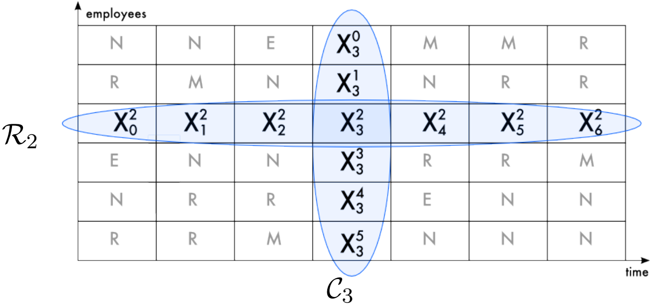
- I set of nurses
- T discrete time horizon *28 days*
- A set of activities *N night, M morning, E evening, R rest*

Nurse Scheduling Problem

- I set of nurses
- T discrete time horizon — *28 days*
- A set of activities — N *night*, M *morning*, E *evening*, R *rest*
- cover constraints $\mathcal{C}_t$ / day t — *between 2 and 3 nurses at night*
- working rules $\mathcal{R}_i$ / nurse i — *at least 2 mornings a week*

Nurse Scheduling Problem

- I set of nurses
- T discrete time horizon — *28 days*
- A set of activities — *N night, M morning, E evening, R rest*
- cover constraints $\mathcal{C}_t$ / day t — *between 2 and 3 nurses at night*
- working rules $\mathcal{R}_i$ / nurse i — *at least 2 mornings a week*



working rules

Examples:

- *between 2 and 3 rests every 7 days*
- *no 3 consecutive nights a week*
- *a rest and a night every week-end*

mutable, heterogeneous, hard/soft

working rules

Examples:

- *between 2 and 3 rests every 7 days*
- *no 3 consecutive nights a week*
- *a rest and a night every week-end*

mutable, heterogeneous, hard/soft

working rules

Examples:

- *between 2 and 3 rests every 7 days*
- *no 3 consecutive nights a week*
- *a rest and a night every week-end*

mutable, heterogeneous, hard/soft

how ?	forbid	enforce	count
what ?	activity	stretch	pattern
when ?	fixed time	sliding period	fixed period

working rules

Examples:

- *between 2 and 3 rests every 7 days*
- *no 3 consecutive nights a week*
- *a rest and a night every week-end*

mutable, heterogeneous, hard/soft

how ?	forbid	enforce	count
what ?	activity	stretch	pattern
when ?	fixed time	sliding period	fixed period

working rules

Examples:

- *between 2 and 3 rests every 7 days*
- *no 3 consecutive nights a week*
- *a rest and a night every week-end*

mutable, heterogeneous, hard/soft

how ?	forbid	enforce	count
what ?	activity	stretch	pattern
when ?	fixed time	sliding period	fixed period

working rules

Examples:

- *between 2 and 3 rests every 7 days*
- *no 3 consecutive nights a week*
- *a rest and a night every week-end*

mutable, heterogeneous, hard/soft

working rules

Examples:

- *between 2 and 3 rests every 7 days*
- *no 3 consecutive nights a week*
- *a rest and a night every week-end*

mutable, heterogeneous, hard/soft

individual constraint penalties (to minimize)

ex: $5 * occurrence(violation)^2$

working rules

Examples:

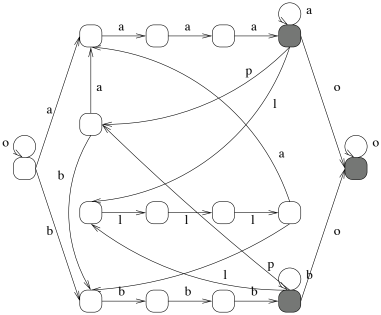
- *between 2 and 3 rests every 7 days*
- *no 3 consecutive nights a week*
- *a rest and a night every week-end*

mutable, heterogeneous, hard/soft

⇒ high-level modelisation tools
⇒ auto-configurable algorithms

Flexible tools in Combinatorial Optimization

finite automata
flexible tool #1



formal languages

- alphabet: Σ a finite non-empty set of symbols

$$\{a, b\}$$

- word/string: $w \in \Sigma^n$ a finite sequence of symbols

$$aaabb$$

- language: $\mathcal{L} \subseteq \Sigma^*$ a set of words

$$\{ab, ba, aab, bba, aaab, bbba, \ldots\}$$

formal languages

- alphabet: Σ a finite non-empty set of symbols

$$\{a, b\}$$

- word/string: $w \in \Sigma^n$ a finite sequence of symbols

$$aaabb$$

- language: $\mathcal{L} \subseteq \Sigma^*$ a set of words

$$\{ab, ba, aab, bba, aaab, bbba, \ldots\}$$

- classes and recognizers: regular, context-free, etc.

formal languages

- alphabet: Σ a finite non-empty set of symbols

$$\{a, b\}$$

- word/string: $w \in \Sigma^n$ a finite sequence of symbols

$$aaabb$$

- language: $\mathcal{L} \subseteq \Sigma^*$ a set of words

$$\{ab, ba, aab, bba, aaab, bbba, \ldots\}$$

- classes and recognizers: regular, context-free, etc.
- operations: union, concatenation, closure, etc.

formal languages

- alphabet: Σ a finite non-empty set of symbols

$$\{a, b\}$$

- word/string: $w \in \Sigma^n$ a finite sequence of symbols

$$aaabb$$

- language: $\mathcal{L} \subseteq \Sigma^*$ a set of words

$$\{ab, ba, aab, bba, aaab, bbba, \ldots\}$$

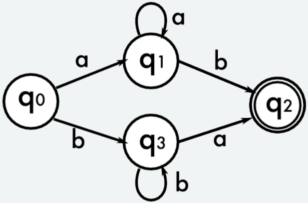
- classes and recognizers: regular, context-free, etc.
- operations: union, concatenation, closure, etc.
- properties: emptiness, membership, universality, etc.

generators and recognizers

$$\mathcal{L} = \{ab, ba, aab, bba, aaab, bbba, \ldots\}$$

1 infinite regular language, 3 finite representations:

finite automaton



regular expression

$$(a^+b)|(b^+a)$$

formal grammar

$$S \rightarrow aA|bB$$
$$A \rightarrow aA|b$$
$$B \rightarrow bB|a$$

what purpose ?

- implicit and concise (finite) representation
- human-readable and machine-processable
- theories and algorithms for operations and decision properties
- models of discrete systems like languages, protocols

what purpose ?

- implicit and concise (finite) representation
- human-readable and machine-processable
- theories and algorithms for operations and decision properties
- models of discrete systems like languages, protocols
- models of working rules
  - alphabet: set of activities $A = \{M, E, N, R\}$
  - word: $w \in A^T$ schedule of an employee
  - language: constrained set of schedules

working rules as a language

working rules as a language

- rule R as a regexp E_R [Pesant04]

  no more than 2 consecutive nights: $E_R = \neg(NNN)$

working rules as a language

- rule R as a regexp E_R [Pesant04]

  no more than 2 consecutive nights: $E_R = \neg(NNN)$
- feasible schedules as a regular language $\mathcal{L}^{\mathcal{R}} \cap A^T$ with

$$\mathcal{L}^{\mathcal{R}} = \bigcap_{R \in \mathcal{R}} \mathcal{L}(E_R) = \mathcal{L}(\neg \bigcup_{R \in \mathcal{R}} \neg E_R)$$

working rules as a language

- rule R as a regexp E_R [Pesant04]

  no more than 2 consecutive nights: $E_R = \neg(NNN)$
- feasible schedules as a regular language $\mathcal{L}^{\mathcal{R}} \cap A^T$ with

$$\mathcal{L}^{\mathcal{R}} = \bigcap_{R \in \mathcal{R}} \mathcal{L}(E_R) = \mathcal{L}(\neg \bigcup_{R \in \mathcal{R}} \neg E_R)$$

- extension to context-free grammars [Sellman06, Quimper06, Côté10]

$$\mathcal{L}(S \to \epsilon, S \to aSb) = \{a^n b^n \mid n \in \mathbb{N}\}$$

working rules as a language

- rule R as a regexp E_R [Pesant04]

  no more than 2 consecutive nights: $E_R = \neg(NNN)$
- feasible schedules as a regular language $\mathcal{L}^{\mathcal{R}} \cap A^T$ with

$$\mathcal{L}^{\mathcal{R}} = \bigcap_{R \in \mathcal{R}} \mathcal{L}(E_R) = \mathcal{L}(\neg \bigcup_{R \in \mathcal{R}} \neg E_R)$$

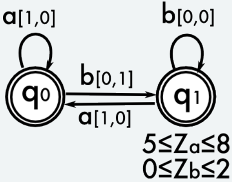
- extension to context-free grammars [Sellman06, Quimper06, Côté10]

$$\mathcal{L}(S \to \epsilon, S \to aSb) = \{a^n b^n \mid n \in \mathbb{N}\}$$

- extension to weighted automata [Demassey05, Menana09]
  for counting, optimization and soft rules

weighted automata

transition costs, path cost, and bounds



- add a vector of costs (index dependent) to each transition
- the cost of the word is the sum of the transition costs
- restrict the language to words with costs within given bounds

working rules as weighted automata [Menana09]

automated modeling tool in ChocoETP

1. model each rule including penalties as a language
   $\Rightarrow$ regex or weighted automaton
2. compute the language intersection
   $\Rightarrow$ multi-weighted automaton

working rules as weighted automata [Menana09]

automated modeling tool in ChocoETP

1. model each rule including penalties as a language
   $\Rightarrow$ regex or weighted automaton
2. compute the language intersection
   $\Rightarrow$ multi-weighted automaton

include parsers for different benchmark formats:

- ASAP3 (XML) www.staffrostersolutions.com
- NRP10 (XML) www.kuleuven-kortrijk.be
- NSPLib (csv) www.projectmanagement.ugent.be
- ETPShoe (csv+txt) [Demassey05]

modeling rules (ex: activity count)

at least one rest on week # 2

- hard rule, 2 alternatives:
  - a regexp $A\{7\}((\neg R)* RA*)A\{14\}$
  - or $A*$ with a counter $Z \in [1, 28]$ and $c_{tR} = 1$ iff $t \in [8, 14]$

modeling rules (ex: activity count)

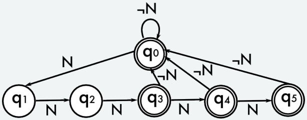
at least one rest on week # 2

- hard rule, 2 alternatives:
  - a regexp $A\{7\}((\neg R)*RA*)A\{14\}$
  - or $A*$ with a counter $Z \in [1, 28]$ and $c_{tR} = 1$ iff $t \in [8, 14]$
- soft rule: (*ex: fixed penalty of 10 if no rest on week 2*)
  $A*$ with a counter $Z \in [0, 28]$ with $c_{tR} = 1$ iff $t \in [8, 14]$
  and an external cost $Y \in [0, 10]$ with $Y = 10 \iff Z < 1$

modeling rules (ex: sliding stretch)

between 3 and 5 consecutive night shifts

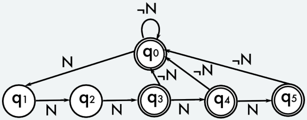
- hard rule:



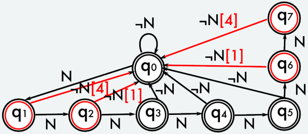
modeling rules (ex: sliding stretch)

between 3 and 5 consecutive night shifts

- hard rule:



- soft rule: (*hard bounds[0, 7] and quadratic penalty*)



with a cost/counter $Y = Z \in [0, +\infty]$

modeling rules (ex: forbid pattern)

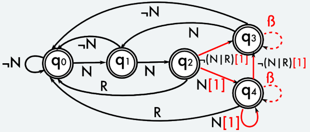
at least one rest after 2 consecutive night shifts

- hard rule:
  - $\neg(A* \, (NN(\neg R)) \, A*)$

modeling rules (ex: forbid pattern)

at least one rest after 2 consecutive night shifts

- hard rule:
  - $\neg(A* \, (NN(\neg R)) \, A*)$
- soft rule: (*ex: linear penalty*)
  1. build the DFA corresponding to $(A* \, (NN(\neg R)\beta*)*)*$
  2. get Q_β the set of states q with outgoing transition β
  3. add a cost $c = 1$ on every ingoing transition of Q_β
  4. associate a cost/counter $Y = Z \in [0, +\infty]$



aggregating rules

satisfying a conjunction of rules

- $R^1 \wedge R^2$ holds iff

$$X \in \mathcal{L}(\Pi^1) \cap \mathcal{L}(\Pi^2) \quad \wedge \quad Z^1 = \sum_t c^1_{tX_t} \quad \wedge \quad Z^2 = \sum_t c^2_{tX_t}$$

aggregating rules

satisfying a conjunction of rules

- $R^1 \wedge R^2$ holds iff
$$X \in \mathcal{L}(\Pi^1) \cap \mathcal{L}(\Pi^2) \quad \wedge \quad Z^1 = \sum_t c^1_{tX_t} \quad \wedge \quad Z^2 = \sum_t c^2_{tX_t}$$
- WFA intersection in the tropical semiring of higher dimension:
$$(\Pi^1, [c^1, 0]) \cap (\Pi^2, [0, c^2]) \in WFA(\Sigma, \mathbb{R}^{n_1+n_2})$$

aggregating rules

satisfying a conjunction of rules

- $R^1 \wedge R^2$ holds iff
$$X \in \mathcal{L}(\Pi^1) \cap \mathcal{L}(\Pi^2) \quad \wedge \quad Z^1 = \sum_t c^1_{tX_t} \quad \wedge \quad Z^2 = \sum_t c^2_{tX_t}$$
- WFA intersection in the tropical semiring of higher dimension:
$$(\Pi^1, [c^1, 0]) \cap (\Pi^2, [0, c^2]) \in WFA(\Sigma, \mathbb{R}^{n_1+n_2})$$

aggregating rules

satisfying a conjunction of rules

- $R^1 \wedge R^2$ holds iff
$$X \in \mathcal{L}(\Pi^1) \cap \mathcal{L}(\Pi^2) \quad \wedge \quad Z^1 = \sum_t c^1_{tX_t} \quad \wedge \quad Z^2 = \sum_t c^2_{tX_t}$$
- WFA intersection in the tropical semiring of higher dimension:
$$(\Pi^1, [c^1, 0]) \cap (\Pi^2, [0, c^2]) \in WFA(\Sigma, \mathbb{R}^{n_1+n_2})$$

(our) intersection algorithm in $WFA(\Sigma, \mathbb{R}^n)$

- convert $WFA(\Sigma, \mathbb{R}^n)$ to $FA(\Sigma \times \mathbb{R}^n)$ and naive intersection modified: $((q_1, q_2), (\sigma_1, \sigma_2), (q'_1, q'_2)) \in \Delta_\cap \iff$
$(q_1, \sigma_1, q'_1) \in \Delta_1 \quad \wedge \quad (q_2, \sigma_2, q'_2) \in \Delta_2 \quad \wedge \quad \text{symbol}(\sigma_1) = \text{symbol}(\sigma_2)$

global constraints
flexible tool #2

constraint satisfaction problem (CSP)

a set of variables $X_1, X_2, \ldots, X_n$
on finite (discrete) domains $D_1, D_2, \ldots, D_n$
related by constraints $C_1, \ldots, C_m$

constraint satisfaction problem (CSP)

a set of variables $X_1, X_2, \ldots, X_n$
on finite (discrete) domains $D_1, D_2, \ldots, D_n$
related by constraints $C_1, \ldots, C_m$

A solution:

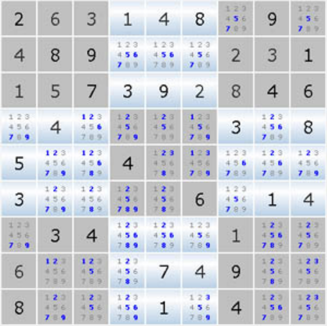
$$(x_1, \ldots, x_n) \in D_1 \times \cdots \times D_n \text{ s.t.}$$
$$C_j(x_1, \ldots, x_n) \text{ holds } \forall j = 1, \ldots, m$$

sudoku as a CSP

$$X_0, X_1, \ldots, X_{80}$$

$$D_i = [0, 9] \quad \forall i \in [0, 80]$$

$$X_0 = 2,\ X_1 = 6,\ \ldots$$
$$X_i \neq X_j \quad \forall (i,j) \in L$$
$$X_i \neq X_j \quad \forall (i,j) \in C$$
$$X_i \neq X_j \quad \forall (i,j) \in S$$



credit: N. Jussien

sudoku as a CSP

$$X_0, X_1, \dots, X_{80}$$

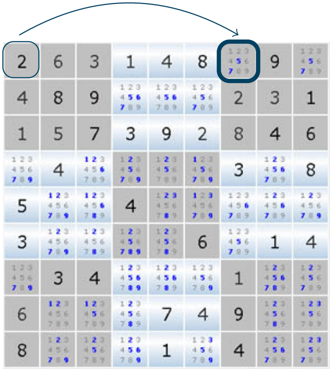
$$D_i = [0,9] \quad \forall i \in [0,80]$$

$$X_0 = 2,\ X_1 = 6,\ \dots$$

$$X_i \neq X_j \quad \forall (i,j) \in L$$

$$X_i \neq X_j \quad \forall (i,j) \in C$$

$$X_i \neq X_j \quad \forall (i,j) \in S$$



arc consistency of $X_0 \neq X_7$: $D_0 = \{2\} \implies$ filter $2 \notin D_7$

backtracking algorithm aka "branch-and-propagate"

1. propagation:
   for each constraint,
   infer inconsistent value assignments
   apply domain reduction
   until fix point
2. tree search:
   if domains are singleton, then solution found
   if no domain is empty, then assign a free variable to a value
   otherwise, backtrack

sudoku as a CSP with global constraints

$X_0, X_1, \ldots, X_{80}$

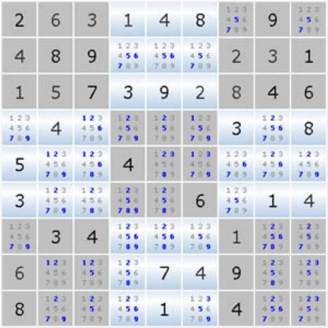
$D_i = [0,9] \quad \forall i \in [0,80]$

$X_0 = 2,\ X_1 = 6,\ \ldots$

$\texttt{alldifferent}(X_i)_{i \in l} \quad \forall l \in L$

$\texttt{alldifferent}(X_i)_{i \in c} \quad \forall c \in C$

$\texttt{alldifferent}(X_i)_{i \in s} \quad \forall s \in S$



sudoku as a CSP with global constraints

$X_0, X_1, \ldots, X_{80}$

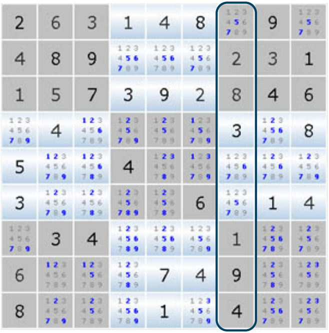
$D_i = [0, 9] \quad \forall i \in [0, 80]$

$X_0 = 2,\ X_1 = 6,\ \ldots$

$\texttt{alldifferent}(X_i)_{i \in l} \quad \forall l \in L$

$\texttt{alldifferent}(X_i)_{i \in c} \quad \forall c \in C$

$\texttt{alldifferent}(X_i)_{i \in s} \quad \forall s \in S$



global AC: $X_{43} \neq 7$

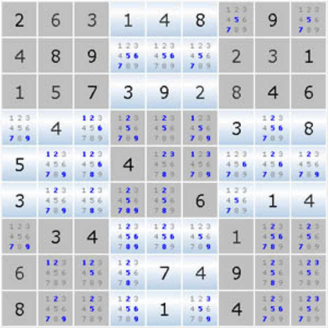
sudoku as a CSP with global constraints

$X_0, X_1, \ldots, X_{80}$

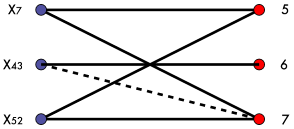
$D_i = [0, 9] \quad \forall i \in [0, 80]$

$X_0 = 2,\ X_1 = 6,\ \ldots$

$\texttt{alldifferent}(X_i)_{i \in l} \quad \forall l \in L$
$\texttt{alldifferent}(X_i)_{i \in c} \quad \forall c \in C$
$\texttt{alldifferent}(X_i)_{i \in s} \quad \forall s \in S$



global AC: $X_{43} \neq 7$
`alldifferent` $\approx$ bipartite matching $O(m\sqrt{n})$ [Régin 94]



examples of value global constraints

- alldifferent$((X_1, X_2, ..., X_n))$ [Régin 94]
- global-cardinality$((X_1, X_2, ..., X_n), (l_j)_j, (u_j)_j)$ [Régin 96]
- among$(Z, (X_1, X_2, ..., X_n), \mathcal{V})$ [Bessière et al. 05]
- soft-alldifferent$(Z, (X_1, X_2, ..., X_n))$ [Petit et al. 01]
- mincost-alldifferent$(Z, (X_1, X_2, ..., X_n), (c_{ij})_{i,j})$ [Sellmann 02]

see also the Global Constraint Catalog http://sofdem.github.io/gccat/

examples of value global constraints

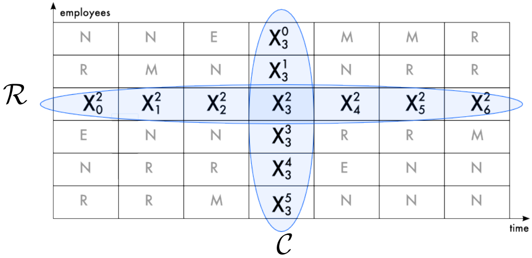
- `alldifferent`$((X_1, X_2, ..., X_n))$ [Régin 94]
- `global-cardinality`$((X_1, X_2, ..., X_n), (l_j)_j, (u_j)_j)$ [Régin 96]
- `among`$(Z, (X_1, X_2, ..., X_n), \mathcal{V})$ [Bessière et al. 05]
- `soft-alldifferent`$(Z, (X_1, X_2, ..., X_n))$ [Petit et al. 01]
- `mincost-alldifferent`$(Z, (X_1, X_2, ..., X_n), (c_{ij})_{i,j})$ [Sellmann 02]

see also the Global Constraint Catalog `http://sofdem.github.io/gccat/`

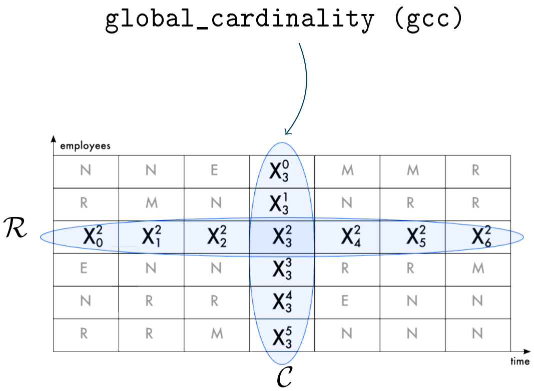
from consistency to filtering

- robustness and incrementality
- level of consistency vs. computation time

a CSP model for NSP

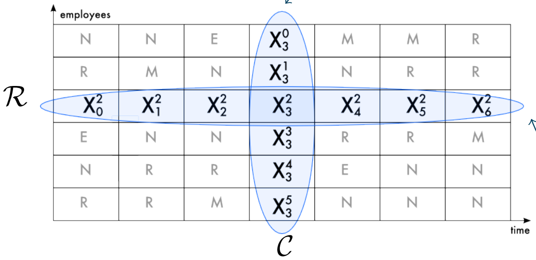


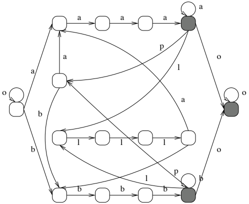
a CSP model for NSP



a CSP model for NSP

global_cardinality (gcc)





language global constraints
flexible solution #1

CSPs as languages

- CSP solution $(x_1, x_2, \ldots, x_n)$ = word $x_1 x_2 \ldots x_n \in D^*$
- CSP model = language representation
- (un)satisfiability = emptiness

CSPs as languages

- CSP solution $(x_1, x_2, \ldots, x_n)$ = word $x_1 x_2 \ldots x_n \in D^*$
- CSP model = language representation
- (un)satisfiability = emptiness

language global constraint family

$$\texttt{language}((X_1, X_2, ..., X_n), \mathcal{L}) \equiv X_1 X_2 \ldots X_n \in \mathcal{L}$$

CSPs as languages

- CSP solution $(x_1, x_2, \ldots, x_n)$ = word $x_1 x_2 \ldots x_n \in D^*$
- CSP model = language representation
- (un)satisfiability = emptiness

language global constraint family

$$\texttt{language}((X_1, X_2, ..., X_n), \mathcal{L}) \equiv X_1 X_2 \ldots X_n \in \mathcal{L}$$

- $\texttt{regular}((X_1, X_2, ..., X_n), \Pi)$ [Pesant 04]
- $\texttt{cost-regular}(Z, (X_1, X_2, ..., X_n), \Pi, c)$ [Demassey 05]
- $\texttt{context-free}((X_1, X_2, ..., X_n), G)$ [Sellman 06, Quimper 06]
- $\texttt{multicost-regular}((Z_1, Z_2, ..., Z_p), (X_1, X_2, ..., X_n), \Pi, c)$ [Menana 09]

language $(< X_1, \ldots, X_n >, \mathcal{L})$

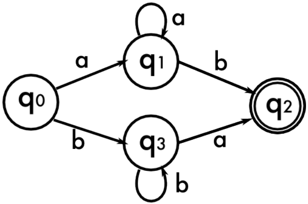
the satisfiability problem

is $\mathcal{L} \cap (D_1 \times \cdots \times D_n)$ empty ?

the consistency problem for $v \in D_i$

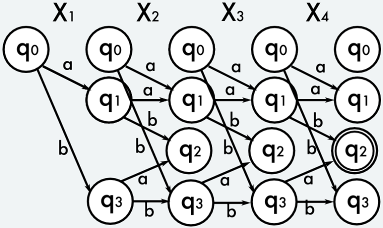
is $\mathcal{L} \cap (D_1 \times \cdots \times D_{i-1} \times \{v\} \times D_{i+1} \times \cdots \times D_n)$ empty ?

regular $(< X_1, \ldots, X_n >, \Pi = (Q, D, \Delta, q_0, F))$



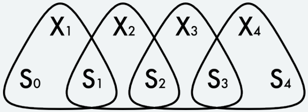
graph connexity [Pesant03]

$$\mathcal{L}(\Pi) \cap (D_1 \times \cdots \times D_n)$$

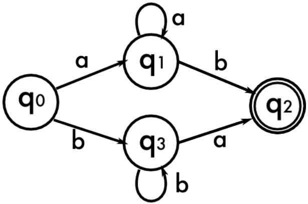


state-decomposition [Beldiceanu04]

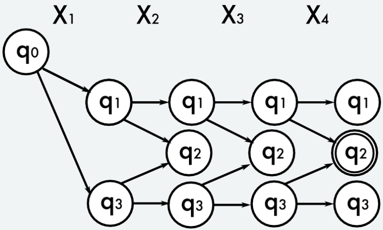
$$\begin{cases} S_i \in Q, & i = 1..n \\ (S_i, X_i, S_{i+1}) \in \Delta, & i = 1..n \end{cases}$$



$\texttt{regular}\ (< X_1, \ldots, X_n >, \Pi = (Q, D, \Delta, q_0, F))$

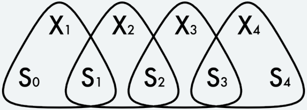


graph connexity [Pesant03]

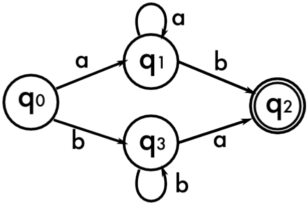


state-decomposition [Beldiceanu04]

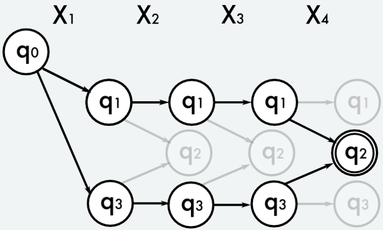
$$\begin{cases} S_i \in Q, & i = 1..n \\ (S_i, X_i, S_{i+1}) \in \Delta, & i = 1..n \end{cases}$$



regular $(< X_1, \ldots, X_n >, \Pi = (Q, D, \Delta, q_0, F))$

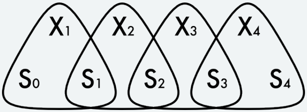


graph connexity [Pesant03]

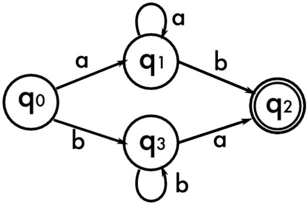


state-decomposition [Beldiceanu04]

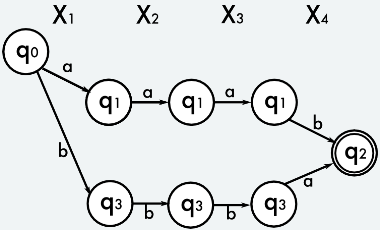
$$\begin{cases} S_i \in Q, & i = 1..n \\ (S_i, X_i, S_{i+1}) \in \Delta, & i = 1..n \end{cases}$$



regular $(< X_1, \ldots, X_n >, \Pi = (Q, D, \Delta, q_0, F))$

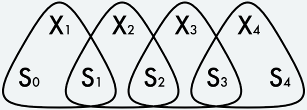


graph connexity [Pesant03]

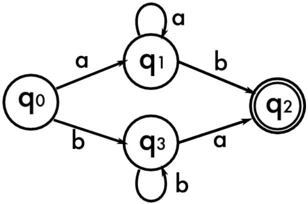


state-decomposition [Beldiceanu04]

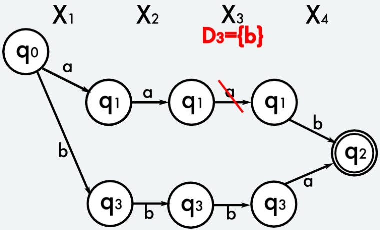
$$\begin{cases} S_i \in Q, & i = 1..n \\ (S_i, X_i, S_{i+1}) \in \Delta, & i = 1..n \end{cases}$$



$\texttt{regular}\ (< X_1, \ldots, X_n >, \Pi = (Q, D, \Delta, q_0, F))$

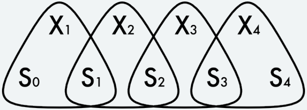


graph connexity [Pesant03]

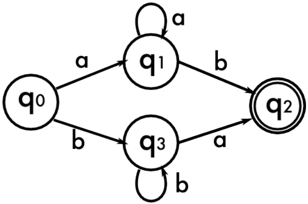


state-decomposition [Beldiceanu04]

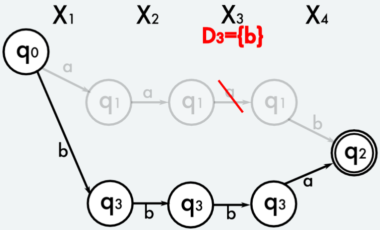
$$\begin{cases} S_i \in Q, & i = 1..n \\ (S_i, X_i, S_{i+1}) \in \Delta, & i = 1..n \end{cases}$$



$\texttt{regular}\ (< X_1, \ldots, X_n >, \Pi = (Q, D, \Delta, q_0, F))$

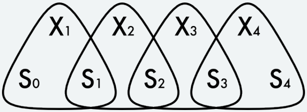


graph connexity [Pesant03]

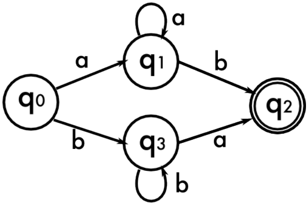


state-decomposition [Beldiceanu04]

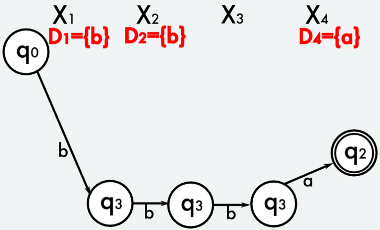
$$\begin{cases} S_i \in Q, & i = 1..n \\ (S_i, X_i, S_{i+1}) \in \Delta, & i = 1..n \end{cases}$$



regular $(< X_1, \ldots, X_n >, \Pi = (Q, D, \Delta, q_0, F))$

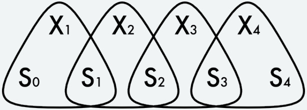


graph connexity [Pesant03]

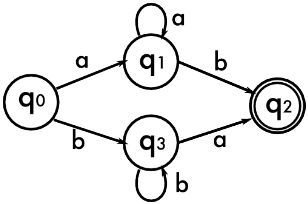


state-decomposition [Beldiceanu04]

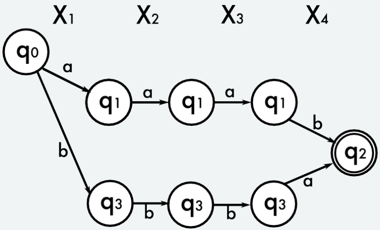
$$\begin{cases} S_i \in Q, & i = 1..n \\ (S_i, X_i, S_{i+1}) \in \Delta, & i = 1..n \end{cases}$$



$\texttt{regular}\,(<X_1,\ldots,X_n>,\Pi=(Q,D,\Delta,q_0,F))$

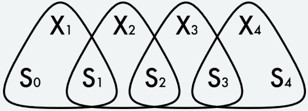


graph connexity [Pesant03]



state-decomposition [Beldiceanu04]

$$\begin{cases} S_i \in Q, & i = 1..n \\ (S_i, X_i, S_{i+1}) \in \Delta, & i = 1..n \end{cases}$$



$O(|\Delta_n|)$ with $|\Delta_n| \ll n|\Delta|$

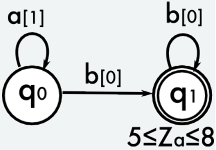
optimization variants

cost-regular$(Z, < X_1, \ldots, X_n >, \Pi, c)$

$$\equiv \; X_1 X_2 \ldots X_n \in \mathcal{L}(\Pi) \; \wedge \; \sum_i c_{iX_i} = Z$$

- shortest/longest path problem
- $O(|\Delta_n|)$ bound consistency on Z

Ilog Solver, Choco [Demassey, Pesant & Rousseau 05]



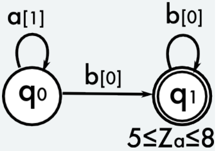
optimization variants

cost-regular$(Z, < X_1, \dots, X_n >, \Pi, c)$

$$\equiv \; X_1 X_2 \dots X_n \in \mathcal{L}(\Pi) \;\wedge\; \textstyle\sum_i c_{iX_i} = Z$$

- shortest/longest path problem
- $O(|\Delta_n|)$ bound consistency on Z

Ilog Solver, Choco [Demassey, Pesant & Rousseau 05]

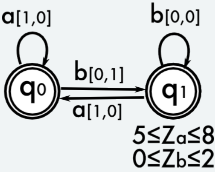


multicost-regular$(< Z^1, \dots, Z^p >, < X_1, \dots, X_n >, \Pi, < c^1, \dots, c^p >)$

$$\equiv \; X_1 X_2 \dots X_n \in \mathcal{L}(\Pi) \;\wedge\; \textstyle\sum_i c^k_{iX_i} = Z^k (\forall k)$$

- resource-constrained SPP/LPP (NP-hard)
- lagrangian relaxation $O(K|\Delta_n|)$

Choco [Menana & Demassey 09]

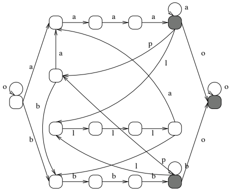


benefit of aggregation (1)

		$\sum$ individual	aggregate	unfolded
full-time	$\#states$	5,782	682	230
	$\#transitions$	40,402	4,768	**400**
part-time	$\#states$	4,401	385	421
	$\#transitions$	30,729	2,689	**681**

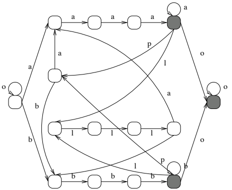
Size of the automata for the `ASAP/GPost` hard instance
for full-time and part-time contracts, $n = 28$

benefit of aggregation (2)



+ assignment costs to minimize
+ cardinality (l, p, o) constraints
1 employee, 96 timeslots
number of working activities $(a, b, ...)$ between 1 and 50
10 instances each
default backtracking of Choco in 10 minutes

benefit of aggregation (2)



+ assignment costs to minimize
+ cardinality (l, p, o) constraints

1 employee, 96 timeslots

number of working activities $(a, b, ...)$ between 1 and 50

10 instances each

default backtracking of Choco in 10 minutes

	multicost-regular			$\bigwedge$ cost-regular			cost-regular $\wedge$ gcc		
$\|A\|$	proof	best	#nodes	proof	best	#nodes	proof	best	#nodes
1	0.0	0.0	41	1.2	1.0	3654	0.3	0.2	225
2	0.1	0.1	68	2.1	0.9	1563	0.6	0.3	393
4	0.2	0.1	67	13.9	8.8	6401	2.9	2.3	1199
8	0.3	0.2	52	101.7	49.8	19637	17.9	13.2	3597
10	0.4	0.4	63	297.2	167.8	44530	50.0	47.7	7615
15	0.8	0.7	63	*50% unsolved*			58.1	47.1	6233
20	1.2	1.0	64	*90% unsolved*			58.1	44.0	4577
30	1.8	1.5	62	*90% unsolved*			*20% unsolved*		
50	5.0	4.8	65	*100% unsolved*			*60% unsolved*		

best = times (s) to find an optimum, proof = time (s) to prove optimality

ChocoETP = DFA + CP + LNS

flexible solution for NSP

a chief nurse-friendly solution ?

1. high-level language to express rules
2. automated tool to model rules
3. automated tool to agregate rules
4. automated tool to solve rules

a chief nurse-friendly solution ?

1. high-level language to express rules
2. automated tool to model rules → WFA/regexp
3. automated tool to agregate rules → WFA intersection
4. automated tool to solve rules → `multicost-regular`

a chief nurse-friendly solution ?

1. high-level language to express rules
2. automated tool to model rules → WFA/regexp
3. automated tool to agregate rules → WFA intersection
4. automated tool to solve rules → `multicost-regular`
5. automated tool to minimize penalties → `CP + LNS`

a chief nurse-friendly solution ?

1. high-level language to express rules
2. automated tool to model rules → WFA/regexp
3. automated tool to agregate rules → WFA intersection
4. automated tool to solve rules → `multicost-regular`
5. automated tool to minimize penalties → `CP + LNS`

ChocoETP

- CP-based Large Neighborhood Search solver
- pluggable parsers
- based on `Choco` and `dk.brics` Java libraries
- `https://github.com/sofdem/chocoETP`

flexibility and effectiveness

hard ASAP instances

		[Métivier 09]	ChocoETP		
	$\|I \times T\|$	cpu	cpu	nodes	bk
Azaiez	13×28	233	**6.3**	4006	5574
Sintef	24×21	-	**1.4**	165	53
Millar-2S-1.1	8×12	1	0.5	29	0
Millar-2S-1	8×12	1	0.3	25	0
Ozkarahan	14×7	1	0.2	24	5

soft ASAP instances

			[Métivier 09]		ChocoETP	
Soft	$\|I \times T\|$	opt	penalty	cpu	penalty	cpu
GPost	8×28	5	8	234	**5**	75
GPost-B	8×28	3	-	-	**3**	3
LLR	27×7	301	**314**	119	320	114
Valouxis	16×28	20	160	3780	**20**	4879
ORTEC01	16×31	270	-	-	**290**	2920

Comparison with an ad-hoc LNS solver [Métivier09]

Conclusion

flexible optimization

modular solutions for recurring problems with mutable constraints

- key of flexibility: decomposed models
- key of effectiveness: aggregated algorithms

flexible optimization

modular solutions for recurring problems with mutable constraints

- key of flexibility: decomposed models
- key of effectiveness: aggregated algorithms

$\Longrightarrow$ automated composition

flexible optimization

modular solutions for recurring problems with mutable constraints

- key of flexibility: decomposed models
- key of effectiveness: aggregated algorithms

$\Longrightarrow$ automated composition $\Longrightarrow$ constraint learning

flexible optimization

modular solutions for recurring problems with mutable constraints

- key of flexibility: decomposed models
- key of effectiveness: aggregated algorithms

$\Longrightarrow$ automated composition $\Longrightarrow$ constraint learning

tools for flexibility

- automata and graphs
- global constraints and propagation
- decomposition methods in linear programming (e.g. [Demassey06])
- linearization (e.g. [Côté13])

Bibliography

- Menana (2011) Automates et programmation par contraintes pour la planification de personnel, PhD Thesis.
- Menana, Demassey (2010) Weighted Automata, Constraint Programming, and Large Neighborhood Search, Nurse Rostering Competition at PATAT'10.
- Menana, Demassey (2009) Sequencing and counting with the multicost-regular constraint, LNCS 5547: 178 - 192.
- Demassey, Pesant, Rousseau (2006) A Cost-Regular based Hybrid Column Generation Approach, Constraints 11 (4) : 315 - 333.
- Demassey, Pesant, Rousseau (2005) Constraint programming based column generation for employee timetabling LNCS 3524: 140 - 154.

Bibliography

- Pesant (2004) A regular language membership constraint for finite sequences of variables, CP'04, 482-495.
- Sellmann (2006) The theory of grammar constraints, CP'06, 530-544.
- Côté, Gendron, Quimper, Rousseau (2007) Formal languages for integer programming modeling of shift scheduling problems, Constraints, 1-23.
- Pesant (2008) Constraint-based rostering, PATAT'08.
- Métivier, Boizumault, Loudni (2009) Solving nurse rostering problems using soft global constraints, CP'09, 73-87.
- Coté, Gendron, Rousseau (2013) Grammar-Based Column Generation for Personalized Multi-Activity Shift Scheduling, INFORMS JoC, 25:3, 461-474.