Constraint Programming and Artificial Intelligence

Challenges, Applications and Opportunities

Barry O'Sullivan

b.osullivan@cs.ucc.ie

[web: http://osullivan.ucc.ie]

Cork Constraint Computation Centre
Department of Computer Science
University College Cork, Ireland

AAAI 2010

UCC
Coláiste na hOllscoile Corcaigh, Éire
University College Cork, Ireland

Dedication

Happy Birthday to my wife Linda and son Daniel. Behind every 'successsful' man there is a rather bemused woman.

Dedication

Happy Birthday to my wife Linda and son Daniel. Behind every ‘successsful’ man there is a rather bemused woman.

Supported by:
Science Foundation Ireland Grant 05/IN/I886.

sfi Fondúireacht Eolaíochta Éireann
Science Foundation Ireland

Credits

The CP and AI Communities

I'm standing on the shoulders of giants.

Colleagues at 4C

James Bowen and **Eugene Freuder**

H.Cambazard, D.Devlin, A.Ferguson, T.Hadzic, E.Hebrard, A.Holland, J.Horan, J.Little, D.Mehta, E.O'Mahony, A.Papadopolous, L.Quesada, H.Simonis, I.Razgon.

Collaborators outside 4C

C.Bessiere, D.Buckley, R.Coletta, B.Faltings, U.Junker, E.Keane, F.Koriche, D.Lesaint, B.Savage, P.Thirion.

Three Messages in this Talk

Challenge

Constraint programming (CP) is powerful, but using it is difficult – can we help reduce the "barrier to entry"?

Application

Most complex real-world applications tackled by CP require the integration of many forms of intelligence.

Opportunity

Artificial intelligence can enable CP. We need you!

Three Messages in this Talk

Challenge

Constraint programming (CP) is powerful, but using it is difficult – can we help reduce the “barrier to entry”?

Application

Most complex real-world applications tackled by CP require the integration of many forms of intelligence.

Opportunity

Artificial intelligence can enable CP. We need you!

Outline of the Talk

The Challenge of Constraint Programming

Ease-of-Use CP – An AI Challenge
- Acquisition
- Reformulation
- Solving

Constraint Problems as Natural Phenomena
- The Importance of Structure
- Parameterised Algorithms

Challenges from Modern Applications

Cancer Treatment Delivery	St.Luke's Hospital, Ireland
Sustainable Harvesting	TreeMetrics, Ireland
Large-scale Energy Management	EDF, France
Energy Efficient Data Centres	EMC Ireland

Wrap-up

Outline

The Challenge of Constraint Programming

Ease-of-Use CP – An AI Challenge
- Acquisition
- Reformulation
- Solving

Constraint Problems as Natural Phenomena
- The Importance of Structure
- Parameterised Algorithms

Challenges from Modern Applications
- Cancer Treatment Delivery — St.Luke's Hospital, Ireland
- Sustainable Harvesting — TreeMetrics, Ireland
- Large-scale Energy Management — EDF, France
- Energy Efficient Data Centres — EMC Ireland

Wrap-up

What is a Constraint Satisfaction Problem?

Variables, Domains and Constraints

Given a set of variables, each taking a value from a domain of possible values, find an assignment to all variables that satisfy the constraints.

- Variables:

- Domain:

- Constraints: Adjacent states must be colored differently

Find a solution!

Combinatorial Optimization is Everywhere

7	5		9		3			6
			4	5				3
6	2			9		8		
	1	5				2	3	
		9		1			7	5
3				8	4			
9			6		1		5	7

(a) Sudoku Puzzle

7	5	8	9	2	3	1	4	6
2	4	3	1	6	7	5	9	8
1	9	6	4	5	8	7	2	3
6	2	7	3	9	5	8	1	4
8	1	5	7	4	6	2	3	9
4	3	9	8	1	2	6	7	5
3	7	1	5	8	4	9	6	2
5	6	4	2	7	9	3	8	1
9	8	2	6	3	1	4	5	7

(b) The Solution

Constraint Modelling Languages

Features

Declarative specification of the problem, separating model formulation, from data, from search strategy.

A Constraint Model of the Sudoku Puzzle

```
matrix = Matrix(N*N,N*N,1,N*N)

sudoku
    = Model( [AllDiff(row) for row in matrix.row],

             [AllDiff(col) for col in matrix.col],

             [AllDiff(matrix[x:x+N, y:y+N].flat)
                for x in range(0,N*N,N)
                for y in range(0,N*N,N)] )
```

Numberjack: `http://numberjack.ucc.ie`

How do we solve a combinatorial problem?

- Polynomial-time Inference, e.g. arc consistency
- Systematic Search, e.g. backtrack search + inference
- Hybrid methods, e.g. operations research with CP
- Satisfiability – CSPs can be translated into CNF
- Local Search – heuristic guess with heuristic repair
- Large Neighbourhood Search – systematic and local search

The AI Challenge

Golomb and Blaumert (JACM 1965)

"... *the success of failure of backtrack (programming) often depends on the skill and ingenuity if the programmer in his ability to adapt the basic methods to the problem at hand and in his ability to reformulate the problem so as to exploit the characteristics of his own computing device.*"

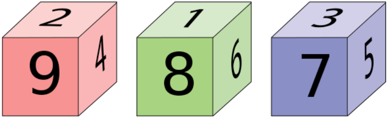
Claim: CP is a rich domain for AI

- CP requires **skill** and **ingenuity**.
- Expertise is difficult to **acquire**, but can we **automate** it?
- Can we provide **assistance** to novices?
- How do we **generalise/transfer** what we've learned?

Why is CP such a challenge?

Example

A set of nontransitive dice is a set of dice for which the relation "*is more likely to roll a higher number*" is not transitive.



Non Transitive Dice - Example

- A solution:

Die A:	1	2	3	4	5	5
Die B:	3	3	3	3	3	3
Die C:	2	2	2	3	6	6

Non Transitive Dice - Example

- A solution:

Die A:	1	2	3	**4**	**5**	**5**
Die B:	**3**	**3**	**3**	**3**	**3**	**3**
Die C:	2	2	2	3	6	6

- A beats B with probability $\frac{1}{2}$ (18 times out of 36).

Non Transitive Dice - Example

- A solution:

Die A:	**1**	**2**	3	4	5	5
Die B:	**3**	**3**	**3**	**3**	**3**	**3**
Die C:	2	2	2	3	6	6

- A beats B with probability $\frac{1}{2}$ (18 times out of 36).
- B beats A with probability $\frac{1}{3}$ (12 times out of 36).

- ***A* beats *B***

Non Transitive Dice - Example

- A solution:

Die A:	1	2	3	4	5	5
Die B:	**3**	**3**	**3**	**3**	**3**	**3**
Die C:	**2**	**2**	**2**	3	6	6

- B beats C with probability $\frac{1}{2}$ (18 times out of 36).
- ***A* beats *B***

Non Transitive Dice - Example

- A solution:

Die A:	1	2	3	4	5	5
Die B:	**3**	**3**	**3**	**3**	**3**	**3**
Die C:	2	2	2	3	**6**	**6**

- B beats C with probability $\frac{1}{2}$ (18 times out of 36).
- C beats B with probability $\frac{1}{3}$ (12 times out of 36).

- ***A* beats *B*, *B* beats *C***

Non Transitive Dice - Example

- A solution:

Die A:	1	2	**3**	4	5	5
Die C:	**2**	**2**	**2**	3	6	6
Die A:	1	2	3	**4**	**5**	**5**
Die C:	**2**	**2**	**2**	**3**	6	6

- A beats C with probability $\frac{5}{12}$ (15 times out of 36).

- ***A* beats *B*, *B* beats *C***

Non Transitive Dice - Example

- A solution:

Die A:	**1**	2	3	4	5	5
Die C:	**2**	**2**	**2**	3	6	6
Die A:	**1**	**2**	3	4	5	5
Die C:	2	2	2	**3**	6	6
Die A:	**1**	**2**	**3**	**4**	**5**	**5**
Die C:	2	2	2	3	**6**	**6**

- A beats C with probability $\frac{5}{12}$ (15 times out of 36).
- C beats A with probability $\frac{17}{36}$ (17 times out of 36).

- ***A* beats *B*, *B* beats *C*, *C* beats *A*!**

Easy to Write a Simple CP Model

```
def beat_count(dice1, dice2):
    return Sum( [ (dice1[i] > dice2[j])
                  for i in range(6)
                  for j in range(6) ] )

def beats(a, b):
    return beat_count(a, b) > beat_count(b, a)


# The Variables
dice_a = VarArray(6, 1, 6, "a_")
dice_b = VarArray(6, 1, 6, "b_")
dice_c = VarArray(6, 1, 6, "c_")

# The Constraints
model = Model( beats( dice_a, dice_b ),
               beats( dice_b, dice_c ),
               beats( dice_c, dice_a ) )

# Solve
solver.solve()
```

Nontransitive Dice: Let's work with the heuristics

Table: Node count for various variable ordering heuristics when finding a set of 6 non-transitive dice with facet values $1, \ldots, 6$.

Heuristic	Basic
Min Dom	–
Min Dom/Deg	–
Min Dom/WDeg	–
Impact	18,206

Nontransitive Dice: Let's work with the model

Symmetries and redundant constraints!

- Order ($\leq$) the values on the face of each dice.
- The smallest dice is lexicographically first.
- The smallest value on the first dice is a 1.

Table: Node count for various variable ordering heuristics when finding a set of 6 non-transitive dice with facet values $1, \ldots, 6$.

Heuristic	Basic	Intra-dice	Inter-dice	Red
Dom	–	–	–	–
Dom/Deg	–	–	–	–
Dom/WDeg	–	9,664	3,346	2,924
Impact	18,206	971	66,053	–

Nontransitive Dice: Let's drive ourselves crazy

Restarting search can often help!

- Restart on the basic model
- Restarts on the 'enhanced' model.

Table: Node count for various variable ordering heuristics when finding a set of 6 non-transitive dice with facet values $1, \ldots, 6$.

Heuristic	Basic	Intra	Inter	Red	RS-B	RS-E.
Dom	–	–	–	–	–	–
Dom/Deg	–	–	–	–	–	–
Dom/WDeg	–	9,664	3,346	**2,924**	14,015	7,341
Impact	18,206	**971**	66,053	–	6,943	**2,572**

The Challenge of CP

Golomb and Blaumert (JACM 1965)

"... *the success of failure of backtrack (programming) often depends on the skill and ingenuity if the programmer in his ability to adapt the basic methods to the problem at hand and in his ability to reformulate the problem so as to exploit the characteristics of his own computing device.*"

...who go on to say...

"That is, backtrack programming ... is somewhat of an art."

Challenges, Applications & Opportunities

Ease-of-Use: An Opportunity for CP and AI

- Acquisition of Constraint Models
- Reformulation of Constraint Models (search + explanability)
- Automated Solver Selection

Changing Perspective: CSPs as Complex Systems

Move away from standard approaches to algorithms and complexity theory $\rightarrow$ typical-case (empirical) and fixed-parameter complexity.

Challenges and Opportunities posed by Applications

Chosen to fit with other invited talks: **cancer** treatment delivery; computational **sustainability**; and **energy** management.

Challenges, Applications & Opportunities

Ease-of-Use: An Opportunity for CP and AI

- Acquisition of Constraint Models
- Reformulation of Constraint Models (search + explanability)
- Automated Solver Selection

Changing Perspective: CSPs as Complex Systems

Move away from standard approaches to algorithms and complexity theory $\rightarrow$ typical-case (empirical) and fixed-parameter complexity.

Challenges and Opportunities posed by Applications

Chosen to fit with other invited talks: **cancer** treatment delivery; computational **sustainability**; and **energy** management.

Challenges, Applications & Opportunities

Ease-of-Use: An Opportunity for CP and AI

- Acquisition of Constraint Models
- Reformulation of Constraint Models (search + explanability)
- Automated Solver Selection

Changing Perspective: CSPs as Complex Systems

Move away from standard approaches to algorithms and complexity theory $\rightarrow$ typical-case (empirical) and fixed-parameter complexity.

Challenges and Opportunities posed by Applications

Chosen to fit with other invited talks: **cancer** treatment delivery; computational **sustainability**; and **energy** management.

Challenges, Applications & Opportunities

Ease-of-Use: An Opportunity for CP and AI

- Acquisition of Constraint Models
- Reformulation of Constraint Models (search + explanability)
- Automated Solver Selection

Changing Perspective: CSPs as Complex Systems

Move away from standard approaches to algorithms and complexity theory $\rightarrow$ typical-case (empirical) and fixed-parameter complexity.

Challenges and Opportunities posed by Applications

Chosen to fit with other invited talks: **cancer** treatment delivery; computational **sustainability**; and **energy** management.

Outline

The Challenge of Constraint Programming

Ease-of-Use CP – An AI Challenge
- Acquisition
- Reformulation
- Solving

Constraint Problems as Natural Phenomena
- The Importance of Structure
- Parameterised Algorithms

Challenges from Modern Applications
- Cancer Treatment Delivery — St.Luke's Hospital, Ireland
- Sustainable Harvesting — TreeMetrics, Ireland
- Large-scale Energy Management — EDF, France
- Energy Efficient Data Centres — EMC Ireland

Wrap-up

My Approach in Part I

I will show how basic AI techniques can be enabling in CP and propose some challenges.

Constraint Acquisition in Context

Where does the model come from?

- We might not have access to a precise statement of the constraints of the problem, e.g. the web, business rules.
- Instead we might have access to an oracle that can provide examples of (non-)solutions to the constraints of the problem, e.g. recommender systems.
- Acquisition can be modelled as a concept learning task.

Concept Learning 101

Concept Learning [Mitchell, 1970s onwards]

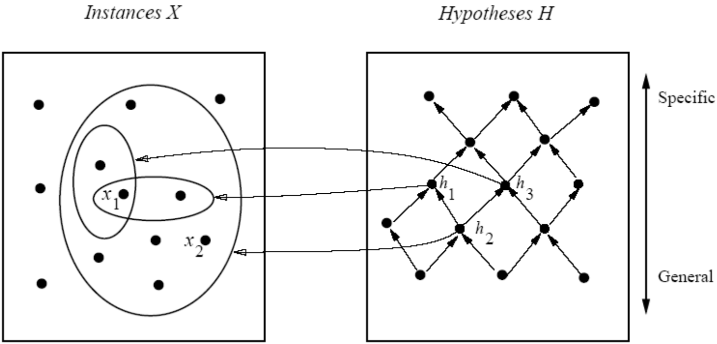
We want to learn a Boolean-valued function.

Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

Question

What is the general concept here?

Ordering over Hypotheses [Mitchell, 1997]

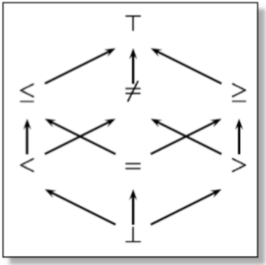


x_1 = <*Sunny, Warm, High, Strong, Cool, Same*>
x_2 = <*Sunny, Warm, High, Light, Warm, Same*>

h_1 = <*Sunny, ?, ?, Strong, ?, ?*>
h_2 = <*Sunny, ?, ?, ?, ?, ?*>
h_3 = <*Sunny, ?, ?, ?, Cool, ?*>

Version Spaces and CSP Acquisition

[Bessiere, Coletta, Koriche, O'Sullivan, CP 2004, ECML 2005]



Hypothesis Space
(Constraint Language)

Target CSP:

V: x_1, x_2 and x_3

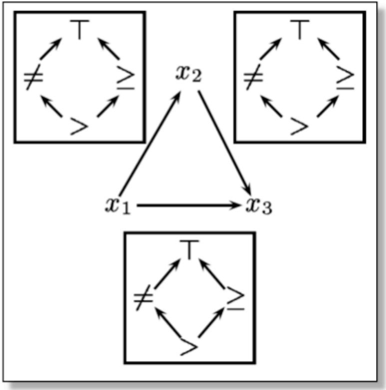
D: $D(x_1) = D(x_2) = D(x_3) = \{1, 2, 3, 4\}$.

C: $\{x_1 > x_2, x_1 > x_3, x_2 > x_3\}$.

Examples:

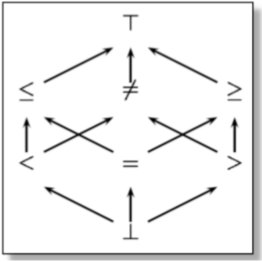
E	x_1	x_2	x_3
e_1^+	4	3	1
e_2^-	2	3	1
e_3^-	3	1	2

Version Spaces and CSP Acquisition

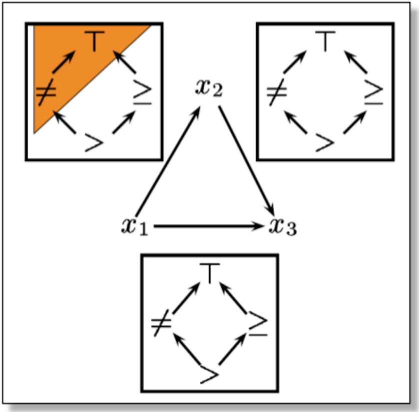


$\{x_1 > x_2, x_1 > x_3, x_2 > x_3\}.$

E	x_1	x_2	x_3
e_1^+	4	3	1
e_2^-	2	3	1
e_3^-	3	1	2

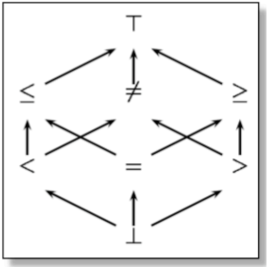


Version Spaces and CSP Acquisition



$\{x_1 > x_2, x_1 > x_3, x_2 > x_3\}.$

E	x_1	x_2	x_3
e_1^+	4	3	1
e_2^-	2	3	1
e_3^-	3	1	2



The ConAcq Algorithm

Algorithm 1: The CONACQ Algorithm

input : a training set (E^+, E^-) and a constraint library B
output: a set of clauses K

$\mathsf{K} \leftarrow \varnothing$
foreach *training instance* e **do**
  $\kappa_e \leftarrow \{\mathsf{b}_{ij} \in \mathsf{B} : e$ does not satisfy $\mathsf{b}_{ij}\}$
  if $e \in E^-$ **then** $\mathsf{K} \leftarrow \mathsf{K} \wedge (\bigvee_{\mathsf{b}_{ij} \in \kappa_e} \mathsf{b}_{ij})$
  if $e \in E^+$ **then** $\mathsf{K} \leftarrow \mathsf{K} \wedge \bigwedge_{\mathsf{b}_{ij} \in \kappa_e} \neg \mathsf{b}_{ij}$
  if `UnitPropagation`(K) *detects* $\perp$ **then** `Return`(*"collapsing"*)

	x_1	x_2	x_3	K
e_1^+	4	3	1	$(\neg \leq_{12}) \wedge (\neg \leq_{13}) \wedge (\neg \leq_{23})$
e_2^-	2	3	1	$(\neg \leq_{12}) \wedge (\neg \leq_{13}) \wedge (\neg \leq_{23}) \wedge (\geq_{12})$
e_3^-	3	1	2	$(\neg \leq_{12}) \wedge (\neg \leq_{13}) \wedge (\neg \leq_{23}) \wedge (\geq_{12}) \wedge (\geq_{23})$

Comment on the Approach

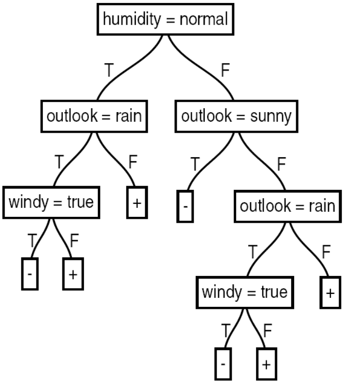
Noise and Soft Constraints

- Noise is a problem – if examples are misclassified our hypothesis space might collapse!
- **Rossi and Sperduti (2001)** – learning soft constraints using reinforcement learning
- **Wilson, et al. (2007)** – interleaving elicitation and solving of CSPs
- **Vu and myself (2007,2008)** – general (soft) constraint acquisition as optimisation

Assumptions

- We assume we have a language to work with (reasonable?);
- We made a big assumption: we know the variables!

Acquisition as Classifier Learning



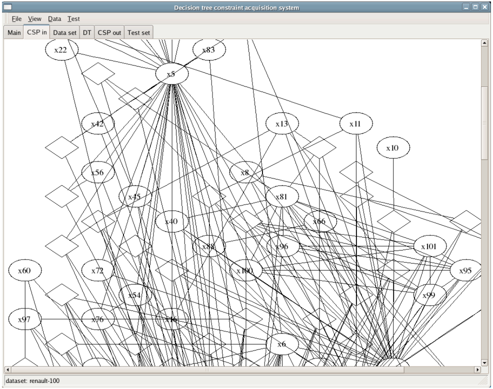
O'Sullivan and Ferguson, IJCAI 2005

Lallouet and Legtchenko, ECML 2005

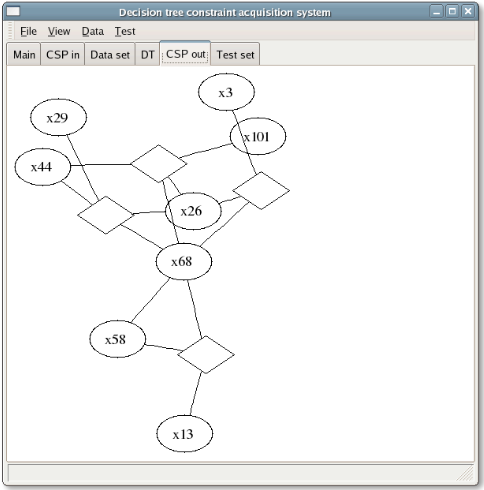
- At each node a test is performed on an attribute.
- We follow a particular path from that node.
- Classifications made at the leaves.
- When classifying assignments to variables in a CSP our classes are "solution" and "non-solution".

Acquisition of a Realworld Configuration Problem

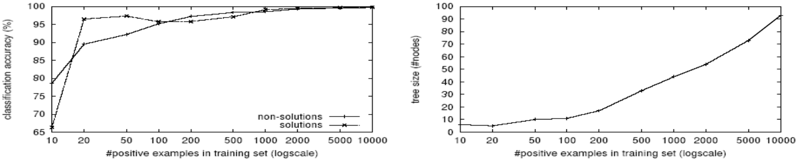
Renault Megane – 100 variables, $\simeq 10^{12}$ solutions!



Acquisition of a Realworld Configuration Problem



Acquisition of a Realworld Configuration Problem



(a) Classification accuracy. (b) Size of the decision tree.

Methodology

- We classified 28,000 solutions and 90,000 non-solutions uniformly distributed throughout the solution space.
- Very small trees, very high classification accuracy (99.9%).
- Tree size is tiny (largest can be stored in a 23kb file) compare with 6.6Mb file to represent the problem, or 3.4Mb if we represent is as an automaton!

AI Challenges in Acquisition

Unsupervised CSP Learning

Unsupervised CSP acquisition, especially in data-rich domains. Data mining has a lot to offer.

Acquiring Problem Classes rather than Instances

We can acquire instances of CSPs, but not the general definition of the general class.

Acquiring Viewpoints

Deciding on the variables, which defines how examples are presented, is an important modelling step.

Model Reformulation

The model is crucially important

- Automatically detect and avoid symmetries;
- Implied constraints;
- Redundant constraints;
- Channelling between models.

Examples of recent work

- Bessiere, Colleta & Petit, CP-05;
- Colton & Miguel, CP-01;
- Charnley, Colton & Miguel, ECAI-06.

Automatic Generation of Implied Constraints

Charnley, Colton & Miguel, ECAI 2005.

An elegant cobination of the following techniques:

Automated Theory Formation

HR performs descriptive learning to speculatively invent concepts and form hypotheses in a domain of interest;

Automated Theorem Proving

Otter uses the resolution method to prove theorems by refutation;

Constraint Logic Programming over Finite Domains

Models are implemented and evaluated using SICStus Prolog.

It works very well in practice

Algebra	Domain size	Basic model	Reformulated model	Proportion (%)
QG3	7	3:09	1:24	44.5
	8	10:07:02	3:10:03	31.3
QG4	6	0:07	0:04	54.2
	7	11:19	5:18	46.8
QG5	7	1:24	1:17	91.7
	8	38:05	28:52	75.8
QG6	9	27:25	6:25	23.4
	10	24:21:00	5:53:03	24.2
QG7	8	19:12	3:33	18.5
	9	27:12:35	4:19:42	15.9
Group	8	16:37	4:15	25.6
	9	4:36:39	28:27	10.3
Moufang	4	0:11	0:08	72.3
	5	10:49	4:19	39.9
Ring	7	0:37	0:30	79.8
	8	4:22	2:09	49.5

Figure: Sample results from [Charnley, Colton & Miguel, ECAI 2005].

Reformulation for Explanation

An Explanation as a Minimal Conflict

Step	Activated constraints					Result	Partial conflict
1.	ρ_1					no fail	$\{\}$
2.	ρ_1	ρ_2				no fail	$\{\}$
3.	ρ_1	ρ_2	ρ_3			no fail	$\{\}$
4.	ρ_1	ρ_2	ρ_3	ρ_4		no fail	$\{\}$
5.	ρ_1	ρ_2	ρ_3	ρ_4	ρ_5	fail	$\{\rho_5\}$
6.	ρ_5					no fail	$\{\rho_5\}$
7.	ρ_5	ρ_1				fail	$\{\rho_1, \rho_5\}$

QuickXplain (Junker, 2004)

As well known family of algorithms due to Junker have been developed for this purpose. A user's preferences over the constraints can be accommodated easily.

Reformulation for Explanation

An Explanation as a Minimal Conflict

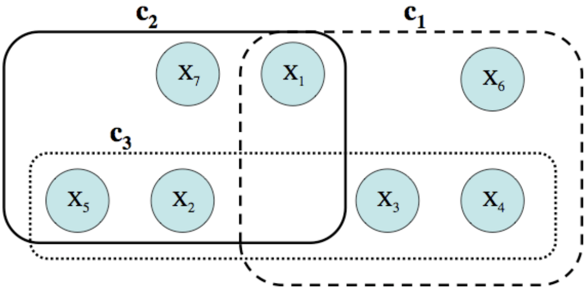
Step	Activated constraints					Result	Partial conflict
1.	ρ_1					no fail	$\{\}$
2.	ρ_1	ρ_2				no fail	$\{\}$
3.	ρ_1	ρ_2	ρ_3			no fail	$\{\}$
4.	ρ_1	ρ_2	ρ_3	ρ_4		no fail	$\{\}$
5.	ρ_1	ρ_2	ρ_3	ρ_4	ρ_5	fail	$\{\rho_5\}$
6.	ρ_5					no fail	$\{\rho_5\}$
7.	ρ_5	ρ_1				fail	$\{\rho_1, \rho_5\}$

QuickXplain (Junker, 2004)

As well known family of algorithms due to Junker have been developed for this purpose. A user's preferences over the constraints can be accommodated easily.

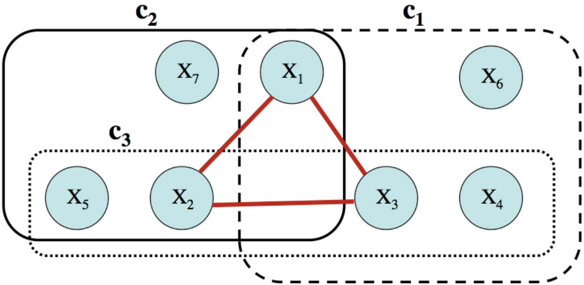
Problem: Algorithms are constraint focused!

Consider a problem defined in terms of three 4-ary constraints



Reformulate to “tighten” the focus

...but it might be possible to reformulate to focus on binary conflicts



Multiple Models for Purpose

We may wish to keep multiple models around – some for solving, others for explanation.

Reformulation using Functional Dependencies

The basic procedure is as follows:

x1	x2	x3	x4
0	0	0	4
0	4	2	4
1	0	0	2
2	2	3	2
2	4	1	3

$\{x1,x2\} \dashrightarrow x4$

x1	x2	x4
0	0	4
0	4	4
1	0	2
2	2	2
2	4	3

x1	x2	x3
0	0	0
0	4	2
1	0	0
2	2	3
2	4	1

x3 --> x2

Normal forms in data base : 3NF, BCNF

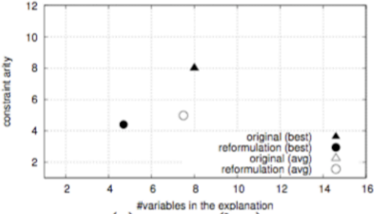
Minimal decomposition

x1	x2	x4
0	0	4
0	2	4
1	0	2
2	2	2
2	4	3

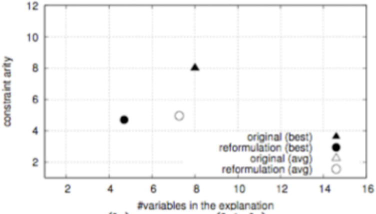
x1	x2
0	0
1	0
2	1
0	2
2	3

x2	x3
0	0
4	1
4	2
2	3

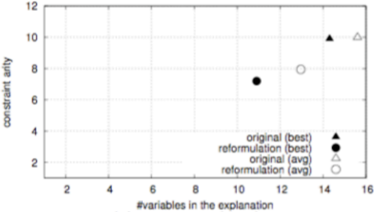
Improved Explanations



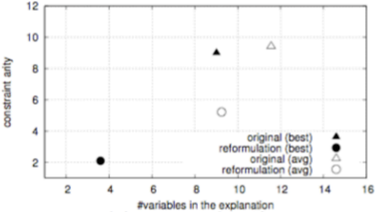
(a) camera (low)



(b) camera (high)



(c) renault (low)



(d) renault (high)

AI Challenges in Reformulation

Viewpoint Identification

Generating candidate sets of variables (model viewpoints) and implied constraints for a problem is extremely difficult.

Identifying Implied Constraints and Structure

Techniques from the fields of machine learning, datamining, and discrete mathematics have the potential to identify interesting candidate implied constraints.

Runtime Prediction – different models and strategies

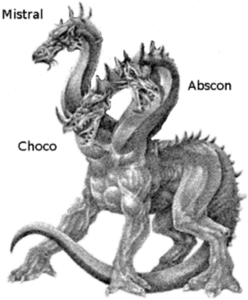
Develop heuristic or learning-based approaches to reliably predict the runtime of a reformulated CSP model using a specific search strategy.

Constraint Solving

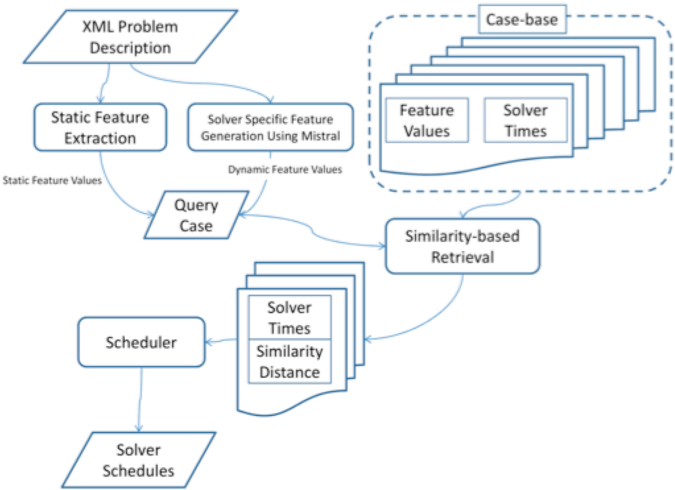
Constraint Solving

CPhydra: a portfolio constraint solver

- Inspired by SATzilla
  – [Xu, et al., JAIR 2008]
- Different solvers.
- International CSP Solver Competition.
- CBR system used to inform the schedule construction based on previously seen problems.
- Schedule optimization system to adapt the CBR results into a schedule of solvers to run.



Architecture of CPHydra – Case-based Reasoning



CPhydra Features

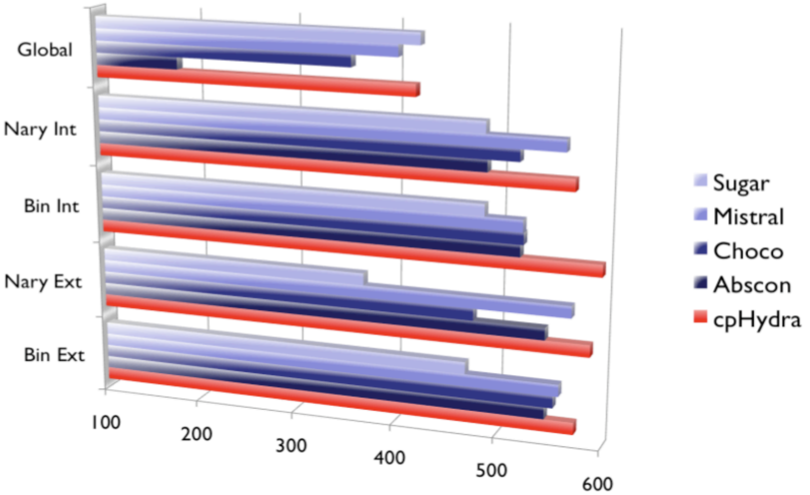
Static Features

- Instances are parsed.
- Numerical features extracted.
- e.g. % extensional constraints etc.

Solver Features

- Mistral run for 2 seconds.
- Features giving information on problem structure from the solver's perspective gathered.
- e.g. Number of checks, number of extra variables.
- Sometimes when gathering the solver features we actually solve the instance.

Performance in the 2008 Competition



AI Challenges in Constraint Solving: Search

Search Advisor Systems

Analyse the key aspects of problem structure and generate advice to novice users on how they should set about solving particular problems.

Robustness in Solver Portfolios

Develop automated search systems that focus on solver objectives such as maximizing the robustness of search time, minimizing the worst-case search time, etc.

Hybrid Solver Generation

Develop tools to support the automated integration of systematic and non-systematic constraint programming methods with operations research techniques.

AI Challenges in Constraint Solving: Inference

Automated Filtering Algorithm Design

Assuming a rule grammar, or primitive constraint language, design a filtering algorithm by searching through the space of possible ‘programs’ in the grammar, evaluating their quality against the specification of the constraint.

Learning When and How to Propagate

Predict the cost (time complexity) and effectiveness (number of pruned values) due to propagating a specific constraint to ensure that global constraints are used in a beneficial manner.

Outline

The Challenge of Constraint Programming

Ease-of-Use CP – An AI Challenge
- Acquisition
- Reformulation
- Solving

Constraint Problems as Natural Phenomena
- The Importance of Structure
- Parameterised Algorithms

Challenges from Modern Applications
- Cancer Treatment Delivery — St.Luke's Hospital, Ireland
- Sustainable Harvesting — TreeMetrics, Ireland
- Large-scale Energy Management — EDF, France
- Energy Efficient Data Centres — EMC Ireland

Wrap-up

CSPs as Naturally Occurring Phenomena

Carla Gomes – we heard from her earlier

CP problems should be studied as naturally occurring phenomena rather than purely mathematical or combinatorial objects.

Why?

Understanding, explaining, and exploiting structure in realworld problems is key to efficiently solving them, e.g. backdoors in satisfiability.

CSPs as Naturally Occurring Phenomena

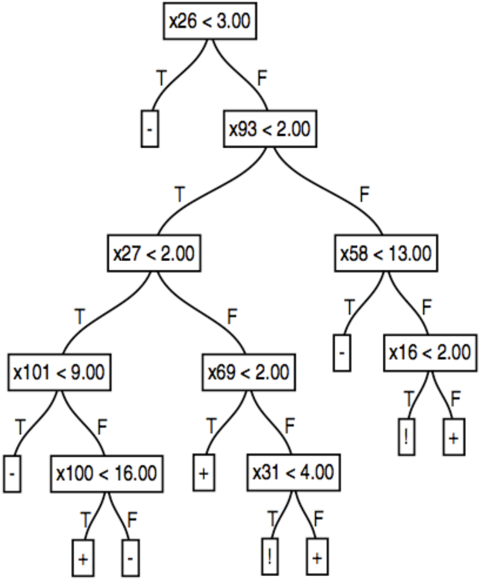
Carla Gomes – we heard from her earlier

CP problems should be studied as naturally occurring phenomena rather than purely mathematical or combinatorial objects.

Why?

Understanding, explaining, and exploiting structure in realworld problems is key to efficiently solving them, e.g. backdoors in satisfiability.

Remember the Large Configuration Problem?



Classification of CSP/SAT

Problem Size Features:
1. **Number of clauses**: denoted c
2. **Number of variables:** denoted v
3. **Ratio:** c/v

Variable-Clause Graph Features:
4-8. **Variable nodes degree statistics:** mean, variation coefficient, min, max and entropy.
9-13. **Clause nodes degree statistics:** mean, variation coefficient, min, max and entropy.

Variable Graph Features:
14-17. **Nodes degree statistics:** mean, variation coefficient, min and max.

Balance Features:
18-20. **Ratio of positive and negative literals in each clause:** mean, variation coefficient and entropy.
21-25. **Ratio of positive and negative occurrences of each variable:** mean, variation coefficient, min, max and entropy.
26-27. **Fraction of binary and ternary clauses**

Proximity to Horn Formula:
28. **Fraction of Horn clauses**
29-33. **Number of occurrences in a Horn clause for each variable:** mean, variation coefficient, min, max and entropy.

DPLL Probing Features:
34-38. **Number of unit propagations:** computed at depths 1, 4, 16, 64 and 256.
39-40. **Search space size estimate:** mean depth to contradiction, estimate of the log of number of nodes.

Local Search Probing Features:
41-44. **Number of steps to the best local minimum in a run:** mean, median, 10th and 90th percentiles for SAPS.
45. **Average improvement to best in a run:** mean improvement per step to best solution for SAPS.
46-47. **Fraction of improvement due to first local minimum:** mean for SAPS and GSAT.
48. **Coefficient of variation of the number of unsatisfied clauses in each local minimum:** mean over all runs for SAPS.

Figure: Features from [Xu, Hutter, Hoos, Leyton-Brown, CP 2007].

Classification of CSP/SAT

Classifier	Class	Crafted Base	Crafted All	Crafted +t	Industrial Base	Industrial All	Industrial +t	Random 3SAT Base	Random 3SAT All	Random 3SAT +t	Random Base	Random All	Random +t
Forest	SAT	78.9	82.5	81.1	93.3	94.1	**94.9**	**98.2**	**99.4**	**99.8**	93.2	96.2	**99.2**
	UNSAT	81.4	83.9	84.4	92.7	92.8	92.9	96.3	**97.2**	99.3	**90.7**	**94.7**	**97.9**
	ALL	**80.5**	83.4	83.1	93.0	**93.5**	94.0	97.1	**98.1**	99.5	92.0	**95.5**	**98.6**
DT	SAT	**82.2**	**84.5**	**83.4**	87.8	89.7	93.3	98.0	97.3	98.0	96.0	95.3	98.5
	UNSAT	78.0	83.5	85.3	**97.1**	**94.7**	93.8	**96.6**	96.8	**99.5**	88.6	93.4	97.5
	ALL	79.3	**83.9**	**84.6**	91.0	91.5	93.5	**97.2**	97.1	**99.6**	**92.3**	94.4	98.0
MLP	SAT	71.8	72.4	71.5	92.6	92.5	95.0	88.4	93.9	88.4	90.9	92.1	**99.2**
	UNSAT	79.4	81.2	79.6	94.9	92.6	**96.3**	90.8	92.4	99.4	82.7	86.7	97.8
	ALL	76.4	77.6	76.4	93.5	92.5	**95.6**	89.8	93.0	99.4	86.8	89.5	98.5
1-NN	SAT	74.7	79.1	78.1	**95.7**	**94.7**	94.6	–	–	–	–	–	–
	UNSAT	80.2	81.7	80.7	94.0	88.6	87.6	–	–	–	–	–	–
	ALL	78.1	80.7	79.8	**95.0**	92.0	91.5	–	–	–	–	–	–
Bayes	SAT	58.6	65.3	69.0	80.2	86.5	87.3	64.2	64.3	87.6	**99.9**	**99.1**	**99.2**
	UNSAT	**84.8**	**85.9**	**85.6**	75.0	76.7	76.9	94.1	93.8	96.9	57.6	56.5	91.7
	ALL	69.4	75.2	69.4	78.1	82.1	82.6	74.9	74.9	92.4	66.0	64.4	95.4

Figure: Classification accuracy for SAT, even industrial problems, can be very high.

Backdoors to Tractability

What is a backdoor in SAT/CSP?

A backdoor to a given problem is a subset of its variables such that, once assigned values, the remaining instance simplifies to a tractable class.

instance	# vars	# clauses	backdoor	fract.
logistics.d	6783	437431	12	0.0018
3bitadd_32	8704	32316	53	0.0061
pipe_01	7736	26087	23	0.0030
qg_30_1	1235	8523	14	0.0113
qg_35_1	1597	10658	15	0.0094

Figure: Backdoors are often small in practice [Gomes et al.].

Fixed Parameter Algorithms

Traditional complexity theory

The running time of an algorithm that solves an NP-Hard problem is exponential in the input size n, e.g. SAT is $\mathcal{O}(2^n)$, unless P=NP.

The fixed parameter algorithm view (informal)

- Downey & Fellows, 1997
- Running time of an algorithm is exponential in a parameter k independent of n;
- Only polynomially dependent on n;

Example

For vertex cover $O(1.3^k + n)$, where k is the maximum number of vertices incident to all edges in the given graph of n vertices.

Cyclic cutset (Feedback Vertex Set)

Problem statement

Input: A CSP Z over n variables
Parameter: The cycle-cutset size, k.
Question: Is it possible to remove at most k variables from Z so that the resulting CSP is acylic.

What do we know?

The problem is FPT and can be solved in time

$$\mathcal{O}(5^k k^2 + poly(n))$$

.

Importance?

A cycle cutset is a backdoor in a binary CSP [Dechter].

Generic Backdoor Computation

Problem statement

Input:	An instance Z of CSP of SAT, a polynomially solvable class P of the given problem.
Parameter:	The size of the backdoor, k
Question:	Is it possible to remove at most k variables from Z so that the resulting instance belongs to P?

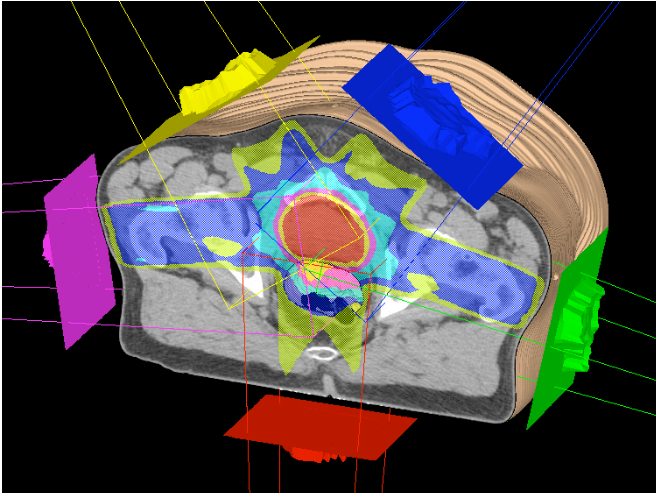
What do we know?

Some classes of this problem are FPT.

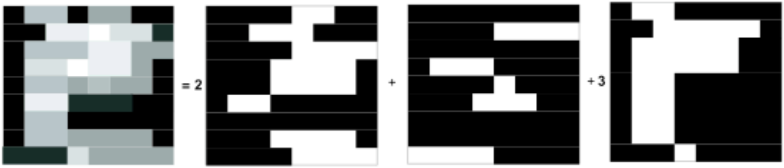
[Razgon and O'Sullivan, Journal of Computer and System Sciences, 2009.]

CP + Parameterised Complexity helps treat Cancer

Delivering Intensity-Modulated Radiation Therapy



Multi-Leaf Collimator Sequencing (Realisation)



$$\begin{bmatrix} 0&3&3&0&2&2&0&0\\ 0&0&5&5&6&4&4&1\\ 0&3&3&3&5&5&2&2\\ 0&4&4&6&5&5&2&0\\ 0&3&3&2&3&2&2&0\\ 0&5&5&1&1&1&0&0\\ 0&3&3&0&0&0&0&0\\ 0&3&3&2&2&2&2&0\\ 1&1&1&4&2&2&2&2 \end{bmatrix} = 2 \begin{bmatrix} 0&0&0&0&1&1&0&0\\ 0&0&1&1&1&0&0&0\\ 0&0&0&0&1&1&1&1\\ 0&0&0&1&1&1&1&0\\ 0&0&0&1&1&1&1&0\\ 0&1&1&0&0&0&0&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&1&1&1&1&0\\ 0&0&0&0&1&1&1&1 \end{bmatrix} + \begin{bmatrix} 0&0&0&0&0&0&0&0\\ 0&0&0&0&1&1&1&1\\ 0&0&0&0&0&0&0&0\\ 0&1&1&1&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&0&0&1&1&1&0&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0\\ 1&1&1&1&0&0&0&0 \end{bmatrix} + 3 \begin{bmatrix} 0&1&1&0&0&0&0&0\\ 0&0&1&1&1&1&1&0\\ 0&1&1&1&1&1&0&0\\ 0&1&1&1&1&1&0&0\\ 0&1&1&0&0&0&0&0\\ 0&1&1&0&0&0&0&0\\ 0&1&1&0&0&0&0&0\\ 0&1&1&0&0&0&0&0\\ 0&0&0&1&0&0&0&0 \end{bmatrix}$$

A fixed parameter complexity result enables us to solve clinical sized instances to optimality.

[Cambazard, O'Mahony, O'Sullivan, CPAIOR 2009].

Global Constraints and Fixed-Parameter Algorithms

NVALUE Constraint

Enforcing domain consistent on NVALUE$([X_1, \ldots, X_n], N)$ is fixed parameter tractable in $k = |\bigcup_{i \in 1 \ldots n} dom(X_i)|$, but is $W[2]$-hard in $k = max(dom(N))$.

DISJOINT Constraint

Enforcing domain consistent on DISJOINT$([X_1, \ldots, X_n], [Y_1, \ldots, Y_m])$ is fixed parameter tractable in $k = |\bigcup_{i \in 1 \ldots n} dom(X_i) \cap \bigcup_{j \in 1 \ldots n} dom(Y_j)|$,

ROOTS Constraint

Enforcing domain consistent on ROOTS$([X_1, \ldots, X_n], S, T)$ is fixed parameter tractable in $k = |ub(T)\ lb(T)$.

[Bessiere et al., AAAI, 2008.]

Challenges in Exploiting Structure

Learn to Identify Backdoors

Can we learn to identify backdoors so that effective heuristic decisions can be made in search?

Deep Understanding of Structure

We've only just begun to fully understand the importance of structure.

Automated Parameter Identification

Can theory formation, automated reasoning, and empirical science be used to identify useful structural parameters?

Outline

The Challenge of Constraint Programming

Ease-of-Use CP – An AI Challenge
- Acquisition
- Reformulation
- Solving

Constraint Problems as Natural Phenomena
- The Importance of Structure
- Parameterised Algorithms

Challenges from Modern Applications

Cancer Treatment Delivery	St.Luke's Hospital, Ireland
Sustainable Harvesting	TreeMetrics, Ireland
Large-scale Energy Management	EDF, France
Energy Efficient Data Centres	EMC Ireland

Wrap-up

60% of all cancer patients will receive radiation therapy

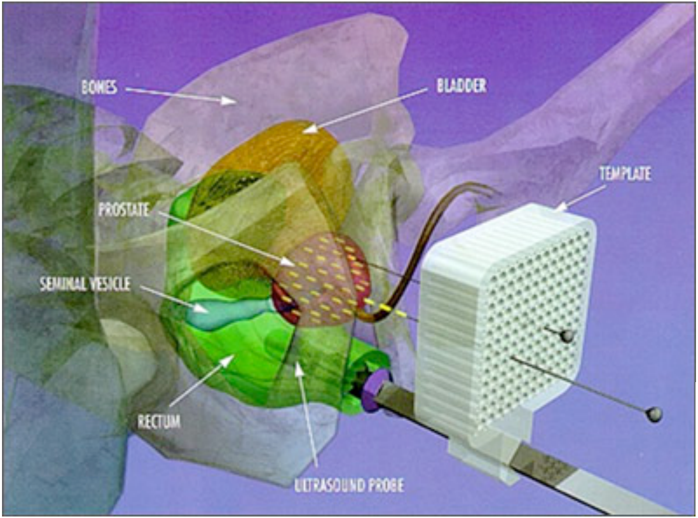
What is it?

- Radiation therapy uses ionising radiation in the treatment of patients diagnosed with cancer.
- The aim of radiation therapy is to deliver a precisely measured dose of radiation to a well-defined tumor volume whilst sparing the surrounding normal tissue.

Variants for which optimisation is key

- **IMRT** we've discussed this earlier.
- **Brachytherapy** involves the placement of radioactive sources directly into tissue.

What is Brachytherapy?



Brachytherapy Treatment Planning - A Model

- The input is a 3-D dosage matrix of lower bounds and upper bounds on radiation exposure.
- Boolean variables encode the (non-)placement of seeds in a 3-D grid of potential locations.
- The treatment plan comprises an assignment to the variables of to represent the (non)placement of seeds in the three-dimensional grid of potential locations.

Brachytherapy Treatment Planning - A Model

- **Eva Lee – Georgia Tech**
- Let $x_i \in \{0, 1\}$ be a variable indicating whether a seed is placed in position i. The total radiation dose at a point P is:

$$\sum_i \delta(||P - X_i||) \cdot x_i$$

where X_i is the coordinates of x_i, and $\delta(d)$ is dose contribution at a distance d (follows an inverse square of distance).

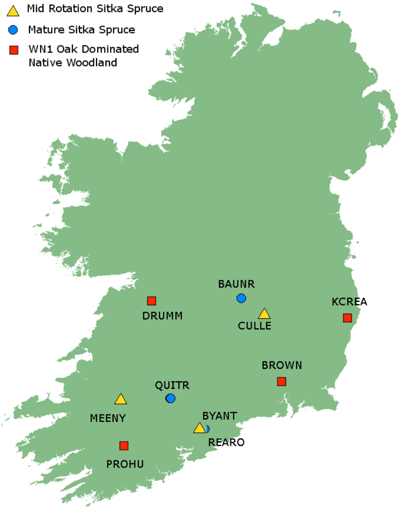
- The basic constraints in the model are:

$$L_p \leq \sum_i \delta(||P - X_i||) \cdot x_i \leq U_p$$

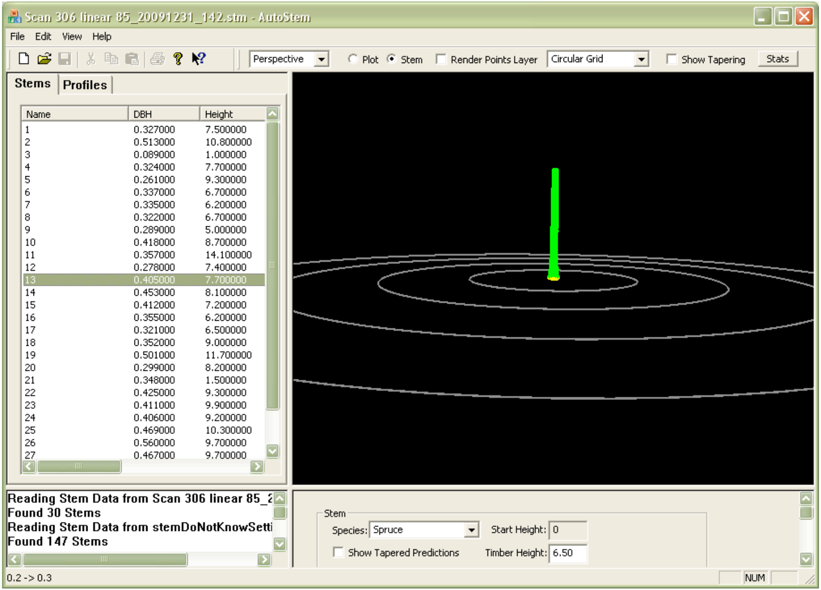
- Maximise the number of point satisfying these constraints.

Sustainable Forest Harvesting TreeMetrics Ltd.

Sustainable Forestry – Biodiversity Awareness



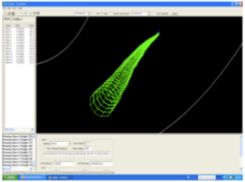
Seeing the Wood from the Trees



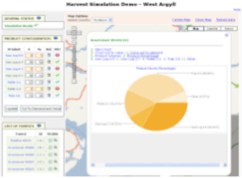
Sustainable Forestry – The Approach

Laser Scan

3D Measurement

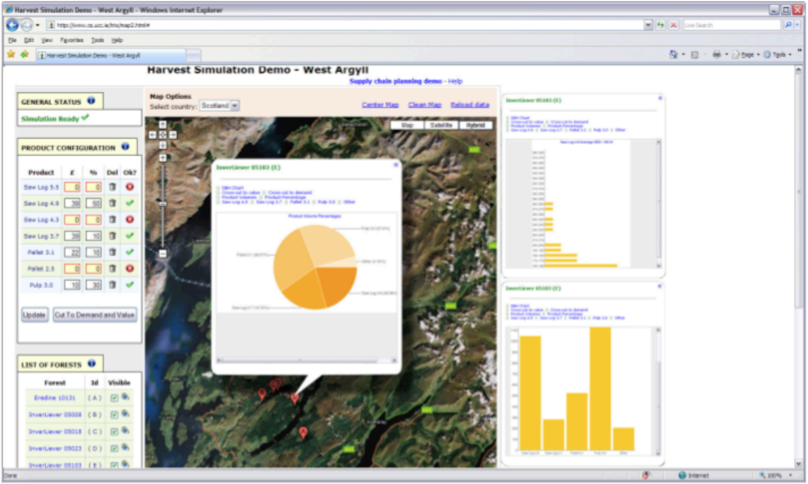


Cutting Software



Optimal Harvest

Sustainable Harvesting – Putting it Together

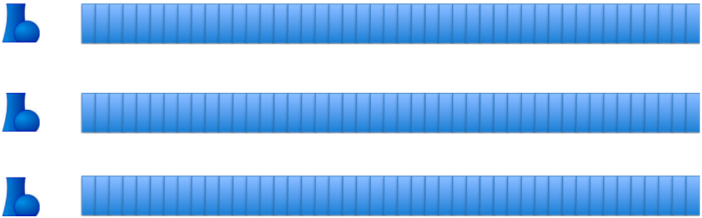


James Little, et al.

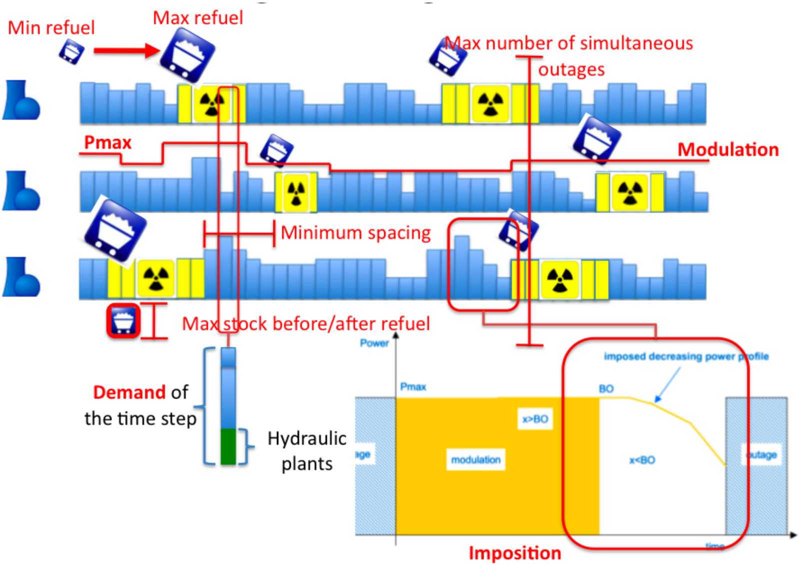
Large-scale Energy Management EDF, France

Large-scale Energy Management EDF, France

- *ROADEF Challenge 2010*
  - Organized by the French Operation Research Society (C. Artigues, E. Bourreau, M. Asfar, E. Ozcan)
  - Proposed by EDF, 44 teams participated worlwide
- Planning the production, refueling and maintenance of thermal powerplants (nuclear or otherwise)



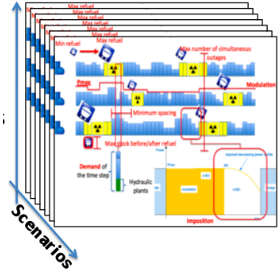
Large-scale Energy Management



Large-scale Energy Management

Data

- 56 Nuclear plants
- 20 Hydraulic plants
- 5800 timesteps
- **121 stochastic scenarios**



Problem size

- More than 50,000,000 *decision* variables, with either continuous or large domains
- Each solution needs about **1 GigaByte** of memory!

Energy Efficient Data Centres EMC Ireland

Energy-Aware Workload Consolidation

- Energy-aware workload management on local data centres based on real-time energy costs.
- Renewable energy management for data centres Ð treating energy storage as an inventory problem.

Energy Cost Minimisation in Networks of Data Centres

- The amortised cost of energy $\simeq 30\%$ of the initial capital investment.
- Exploit energy cost per unit of computation between multiple locations.

Outline

The Challenge of Constraint Programming

Ease-of-Use CP – An AI Challenge
- Acquisition
- Reformulation
- Solving

Constraint Problems as Natural Phenomena
- The Importance of Structure
- Parameterised Algorithms

Challenges from Modern Applications
- Cancer Treatment Delivery — St.Luke's Hospital, Ireland
- Sustainable Harvesting — TreeMetrics, Ireland
- Large-scale Energy Management — EDF, France
- Energy Efficient Data Centres — EMC Ireland

Wrap-up

Let's Sum Up

The Challenge of Constraint Programming

Ease-of-Use CP – An AI Challenge

- Acquisition
- Reformulation
- Solving

Constraint Problems as Natural Phenomena

- The Importance of Structure
- Parameterised Algorithms

Challenges from Modern Applications

Cancer Treatment Delivery	St.Luke's Hospital, Ireland
Sustainable Harvesting	TreeMetrics, Ireland
Large-scale Energy Management	EDF, France
Energy Efficient Data Centres	EMC Ireland

Wrap-up

Constraint Programming and Artificial Intelligence

Challenges, Applications and Opportunities

Barry O'Sullivan

b.osullivan@cs.ucc.ie

[web: http://osullivan.ucc.ie]

Cork Constraint Computation Centre
Department of Computer Science
University College Cork, Ireland

AAAI 2010