Efficient Generic Search Heuristics within the EMBP framework

R. Le Bras[1,2] A. Zanarini[1,2] G. Pesant[1,2]

[1]École Polytechnique de Montréal, Montreal, Canada

[2]CIRRELT, Université de Montréal, Montreal, Canada
{Ronan.LeBras,Alessandro.Zanarini,Gilles.Pesant}@cirrelt.ca

Constraint Programming, September 20-24, 2009

The QWH problem

4	3	5	1	2
1	5	4	2	3
5	2	1	3	4
2	1	3	4	5
3	4	2	5	1

Definition

A *Latin Square* is

- an $m \times m$ grid
- where every cell takes a value in $\{1 \ldots m\}$
- such that two cells in a same row (or column) take distinct values.

The QWH problem

4				2
			2	
	2			
2	1	3	4	5
	4	2	5	

Definition

A *Quasigroup With Holes* (QWH) is

- a partially filled *Latin Square*.

Finding a solution

4				2
			2	
	2			
2	1	3	4	5
	4	2	5	

Iteratively, we:

- enforce domain consistency on the $2m$ constraints
- face a branching decision problem

Finding a solution

4	3,5	1,5	1,3	2
1,3,5	3,5	1,4,5	2	1,3,4
1,3,5	2	1,4,5	1,3	1,3,4
2	1	3	4	5
1,3	4	2	5	1,3

Iteratively, we:

- enforce domain consistency on the $2m$ constraints
- face a branching decision problem

Finding a solution

4				2
			2	
	2	?		
2	1	3	4	5
	4	2	5	

Iteratively, we:

- enforce domain consistency on the $2m$ constraints
- face a branching decision problem

Defining search heuristics

What if we could have a quick glance at the solution set?

4				2
			2	
	2			
2	1	3	4	5
	4	2	5	

4	3	5	1	2
1	5	4	2	3
5	2	1	3	4
2	1	3	4	5
3	4	2	5	1

4	3	5	1	2
3	5	1	2	4
5	2	4	3	1
2	1	3	4	5
1	4	2	5	3

4	3	5	1	2
3	5	4	2	1
5	2	1	3	4
2	1	3	4	5
1	4	2	5	3

4	5	1	3	2
1	3	5	2	4
5	2	4	1	3
2	1	3	4	5
3	4	2	5	1

4	5	1	3	2
5	3	4	2	1
3	2	5	1	4
2	1	3	4	5
1	4	2	5	3

Defining search heuristics

4				2
			2	
	2	x_{13}		
2	1	3	4	5
	4	2	5	

Then, if we were enumerating all the solutions, we could deduce:

- $$P(x_{13}) = \begin{cases} P(x_{13} = 1) = 2/5 \\ P(x_{13} = 4) = 2/5 \\ P(x_{13} = 5) = 1/5 \end{cases}$$

Instead, we try to approximate this probability distribution:

- $$\theta_{x_{13}} = \begin{cases} \theta_{x_{13}}(1) = 0.391 \\ \theta_{x_{13}}(4) = 0.472 \\ \theta_{x_{13}}(5) = 0.137 \end{cases}$$

Defining search heuristics

4				2
			2	
	2	x_{13}		
2	1	3	4	5
	4	2	5	

Then, if we were enumerating all the solutions, we could deduce:

- $P(x_{13}) = \begin{cases} P(x_{13} = 1) = 2/5 \\ P(x_{13} = 4) = 2/5 \\ P(x_{13} = 5) = 1/5 \end{cases}$

Instead, we try to approximate this probability distribution:

- $\theta_{x_{13}} = \begin{cases} \theta_{x_{13}}(1) = 0.391 \\ \theta_{x_{13}}(4) = 0.472 \\ \theta_{x_{13}}(5) = 0.137 \end{cases}$

Outline

1. Expectation-Maximization Belief-Propagation
   - Definitions
   - EMBP framework
   - Computing EMBP
2. EMBP and local consistency
   - EMBPa for the `alldifferent` constraint
   - EMBP-Lsup
3. EMBP and global consistency
   - EMBP-Gsup
4. Experiments
   - Using Θ within a variable-value ordering
   - Pool of heuristics
   - Results

Outline

1. Expectation-Maximization Belief-Propagation
   - Definitions
   - EMBP framework
   - Computing EMBP
2. EMBP and local consistency
   - EMBPa for the `alldifferent` constraint
   - EMBP-Lsup
3. EMBP and global consistency
   - EMBP-Gsup
4. Experiments
   - Using Θ within a variable-value ordering
   - Pool of heuristics
   - Results

Outline

1. Expectation-Maximization Belief-Propagation
   - Definitions
   - EMBP framework
   - Computing EMBP
2. EMBP and local consistency
   - EMBPa for the `alldifferent` constraint
   - EMBP-Lsup
3. EMBP and global consistency
   - EMBP-Gsup
4. Experiments
   - Using Θ within a variable-value ordering
   - Pool of heuristics
   - Results

Outline

1. Expectation-Maximization Belief-Propagation
   - Definitions
   - EMBP framework
   - Computing EMBP
2. EMBP and local consistency
   - EMBPa for the `alldifferent` constraint
   - EMBP-Lsup
3. EMBP and global consistency
   - EMBP-Gsup
4. Experiments
   - Using Θ within a variable-value ordering
   - Pool of heuristics
   - Results

Outline

Definition

A *Constraint Satisfaction Problem* (CSP) consists of

1. a finite set of variables $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$
2. with finite domains $\mathcal{D} = \{D_1, D_2, \ldots, D_n\}$ such that $x_i \in D_i$ for all i
3. together with a finite set of constraints $\mathcal{C} = \{C_1, C_2, \ldots, C_n\}$, each on a subset of $X(C_i) \subseteq \mathcal{X}$.

Notation

$$
\begin{aligned}
d &= \max_{x_i \in \mathcal{X}} |D_i| \\
m &= \max_{C_i \in \mathcal{C}} |X(C_i)| \\
k &= \max_{x_i \in \mathcal{X}} |\{C_j : x_i \in X(C_j)\}|
\end{aligned}
$$

Definition

A *Constraint Satisfaction Problem* (CSP) consists of

1. a finite set of variables $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$
2. with finite domains $\mathcal{D} = \{D_1, D_2, \ldots, D_n\}$ such that $x_i \in D_i$ for all i
3. together with a finite set of constraints $\mathcal{C} = \{C_1, C_2, \ldots, C_n\}$, each on a subset of $X(C_i) \subseteq \mathcal{X}$.

Notation

$$\begin{aligned} d &= \max_{x_i \in \mathcal{X}} |D_i| \\ m &= \max_{C_i \in \mathcal{C}} |X(C_i)| \\ k &= \max_{x_i \in \mathcal{X}} |\{C_j : x_i \in X(C_j)\}| \end{aligned}$$

Definition

The EMBP framework introduces the following definitions:

Θ	Probability distribution of the variables
y	Binary-vector random variable indicating whether the constraints are satisfied
z	Satisfying configurations of the constraints

Essentially, EMBP aims at maximizing the loglikelihood of $P(y, z|\Theta)$.

Intuition

We observe a solution of the CSP (y) but we do not really know how the constraints are satisfied (z). Thus, we are asking EMBP to figure out how the variables are set (Θ).

Definition

The EMBP framework introduces the following definitions:

Θ	Probability distribution of the variables
y	Binary-vector random variable indicating whether the constraints are satisfied
z	Satisfying configurations of the constraints

Essentially, EMBP aims at maximizing the loglikelihood of $P(y, z|\Theta)$.

Intuition

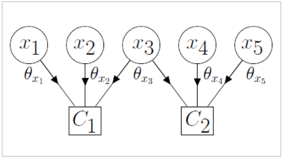
We observe a solution of the CSP (y) but we do not really know how the constraints are satisfied (z). Thus, we are asking EMBP to figure out how the variables are set (Θ).

Additional definition

Let $Q(z)$ be the distribution function $P(z|y,\Theta)$, representing each solution probability given the biases Θ and given the observation y that the constraints are satisfied.

EMBP *(Hsu et al.)* iteratively adjusts Θ in a two-step process:

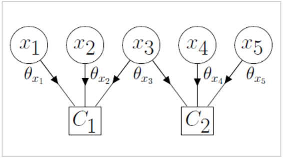
EMBP *(Hsu et al.)* iteratively adjusts Θ in a two-step process:



(a) E-Step

In the E-Step, we compute the probability of the satisfying configurations given the variable distribution:
$Q(z) = \prod_{i=1}^{k}(q(C_i))$,
where $q(C_i)$ is the probability of a given configuration for C_i.

EMBP *(Hsu et al.)* iteratively adjusts Θ in a two-step process:

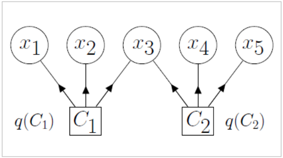


(a) E-Step

In the E-Step, we compute the probability of the satisfying configurations given the variable distribution:

$Q(z) = \prod_{i=1}^{k}(q(C_i))$,

where $q(C_i)$ is the probability of a given configuration for C_i.



(b) M-Step

In the M-Step, the variables adjust their distribution taking into account the probability of the valid tuples of the constraints:

$\theta_{x_i}(v) = \frac{1}{\eta} \sum_{C_k \in \mathcal{C}: x_i \in X(C_k)} \left(\sum_{z \in S_z : x_i = v} Q(z) \right)$

Computing EMBP: a tradeoff between accuracy and complexity

Within the EMBP framework, the definition of $Q(z)$ remains however unspecified.

Computing $Q(z)$ exactly implies expressing every single solution of the problem, which is clearly intractable.

Hence, we approximate $Q(z)$ and the following methods gradually improve the accuracy of the estimation of $Q(z)$.

Computing EMBP: a tradeoff between accuracy and complexity

Within the EMBP framework, the definition of $Q(z)$ remains however unspecified.

Computing $Q(z)$ exactly implies expressing every single solution of the problem, which is clearly intractable.

Hence, we approximate $Q(z)$ and the following methods gradually improve the accuracy of the estimation of $Q(z)$.

Computing EMBP: a tradeoff between accuracy and complexity

Within the EMBP framework, the definition of $Q(z)$ remains however unspecified.

Computing $Q(z)$ exactly implies expressing every single solution of the problem, which is clearly intractable.

Hence, we approximate $Q(z)$ and the following methods gradually improve the accuracy of the estimation of $Q(z)$.

Outline

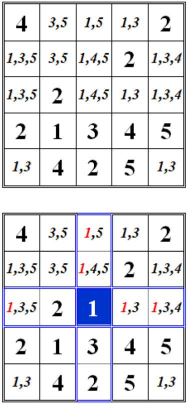
Within the *alldifferent* constraint, the probability that variable x_i is assigned the value v can be approximated by the probability that no other variable in the constraint takes the value v.

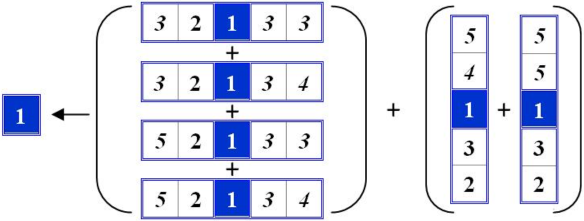
Within the *alldifferent* constraint, the probability that variable x_i is assigned the value v can be approximated by the probability that no other variable in the constraint takes the value v.

4	3,5	1,5	1,3	2
1,3,5	3,5	1,4,5	2	1,3,4
1,3,5	2	1,4,5	1,3	1,3,4
2	1	3	4	5
1,3	4	2	5	1,3

4	3,5	1,5	1,3	2
1,3,5	3,5	1,4,5	2	1,3,4
1,3,5	2	1	1,3	1,3,4
2	1	3	4	5
1,3	4	2	5	1,3

Within the *alldifferent* constraint, the probability that variable x_i is assigned the value v can be approximated by the probability that no other variable in the constraint takes the value v.





Theoretically, we add $\mathcal{O}(k.d^m)$ terms together.

Within the *alldifferent* constraint, the probability that variable x_i is assigned the value v can be approximated by the probability that no other variable in the constraint takes the value v.

4	3,5	1,5	1,3	2
1,3,5	3,5	1,4,5	2	1,3,4
1,3,5	2	1,4,5	1,3	1,3,4
2	1	3	4	5
1,3	4	2	5	1,3

$$\theta_{x_i}(v) = \frac{1}{\eta} \sum_{C_k \in \mathcal{C} : x_i \in X(C_k)} \left(\prod_{x_j \in X(C) \setminus x_i} (1 - \theta_{x_j}(v)) \right)$$

4	3,5	1,5	1,3	2
1,3,5	3,5	1,4,5	2	1,3,4
1,3,5	2	1	1,3	1,3,4
2	1	3	4	5
1,3	4	2	5	1,3

where η is a normalizing constant.

(*Hsu et al., 2007*)

Complexity: $\mathcal{O}(k.m)$, i.e. $\mathcal{O}(2.\sqrt{n})$ for the QWH.

First Contribution

We propose to derive local X-consistency EMBP methods, which are a fairly natural extension to EMBPa, to improve the accuracy of EMBPa and extend this approach to any constraint.

We consider all assignments $y = b$ that are X-consistent with the assignment $x = a$ within a constraint to compute $\theta_x(a)$ for a given constraint.

We consider all assignments $y = b$ that are X-consistent with the assignment $x = a$ within a constraint to compute $\theta_x(a)$ for a given constraint.

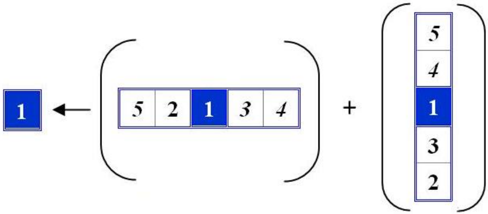
4	3,5	1,5	1,3	2
1,3,5	3,5	1,4,5	2	1,3,4
1,3,5	2	1,4,5	1,3	1,3,4
2	1	3	4	5
1,3	4	2	5	1,3

4	3,5	1,5	1,3	2
1,3,5	3,5	1,4,5	2	1,3,4
1,3,5	2	1	1,3	1,3,4
2	1	3	4	5
1,3	4	2	5	1,3

We consider all assignments $y = b$ that are X-consistent with the assignment $x = a$ within a constraint to compute $\theta_x(a)$ for a given constraint.

4	3,5	1,5	1,3	2
1,3,5	3,5	1,4,5	2	1,3,4
1,3,5	2	1,4,5	1,3	1,3,4
2	1	3	4	5
1,3	4	2	5	1,3

4	3,5	1,5	1,3	2
1,3,5	3,5	1,4,5	2	1,3,4
1,3,5	2	1	1,3	1,3,4
2	1	3	4	5
1,3	4	2	5	1,3



Theoretically, we add $\mathcal{O}(k.d^m)$ terms together.

We consider all assignments $y = b$ that are X-consistent with the assignment $x = a$ within a constraint to compute $\theta_x(a)$ for a given constraint.

4	3,5	1,5	1,3	2
1,3,5	3,5	1,4,5	2	1,3,4
1,3,5	2	1,4,5	1,3	1,3,4
2	1	3	4	5
1,3	4	2	5	1,3

$$\theta_{x_i}(v) = \frac{1}{\eta} \sum_{C_k \in \mathcal{C} : x_i \in X(C_k)} \left(\prod_{x_j \in X(C_k) \setminus x_i} \sum_{v' \in \tilde{D}_{x_i = v}(x_j)} \theta_{x_j}(v') \right)$$

4	3,5	1,5	1,3	2
1,3,5	3,5	1,4,5	2	1,3,4
1,3,5	2	1	1,3	1,3,4
2	1	3	4	5
1,3	4	2	5	1,3

where $\tilde{D}_{x_i=v}(x_j)$ represents the reduced domain of the variable x_j after assigning $x_i = v$ and enforcing X-consistency on C_i.

Complexity: $\mathcal{O}(k.m.d)$, i.e. $\mathcal{O}(2.n)$ for the QWH.

Advantages

- Better accuracy with a stronger consistency
- Generic, thus easily implementable for any constraint

Drawbacks

- Higher complexity, due to the propagation at the constraint level

Advantages

- Better accuracy with a stronger consistency
- Generic, thus easily implementable for any constraint

Drawbacks

- Higher complexity, due to the propagation at the constraint level

Outline

Second Contribution

We suggest to go one step further in terms of accuracy for the computation of $Q(z)$.

With EMBP-Gsup, the problem is considered as a whole and the method directly exploits the dependence between constraints when computing $Q(z)$.

EMBP-Gsup improves the quality of the approximation taking into account supports that are X consistent after propagating every constraint of the problem.

We consider all assignments $y = b$ that are X-consistent with the assignment $x = a$ within the whole problem to compute $\theta_x(a)$ for a given constraint.

We consider all assignments $y = b$ that are X-consistent with the assignment $x = a$ within the whole problem to compute $\theta_x(a)$ for a given constraint.

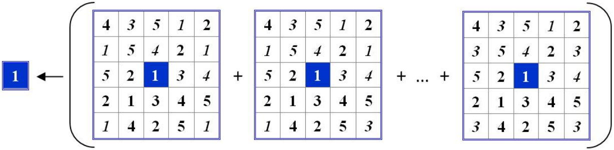
4	3,5	1,5	1,3	2
1,3,5	3,5	1,4,5	2	1,3,4
1,3,5	2	1,4,5	1,3	1,3,4
2	1	3	4	5
1,3	4	2	5	1,3

4	3,5	1,5	1,3	2
1,3,5	3,5	1,4,5	2	1,3,4
1,3,5	2	1	1,3	1,3,4
2	1	3	4	5
1,3	4	2	5	1,3

We consider all assignments $y = b$ that are X-consistent with the assignment $x = a$ within the whole problem to compute $\theta_x(a)$ for a given constraint.

4	3,5	1,5	1,3	2
1,3,5	3,5	1,4,5	2	1,3,4
1,3,5	2	1,4,5	1,3	1,3,4
2	1	3	4	5
1,3	4	2	5	1,3

4	3,5	1,5	1,3	2
1,3,5	3,5	1,4,5	2	1,3,4
1,3,5	2	1	1,3	1,3,4
2	1	3	4	5
1,3	4	2	5	1,3



Theoretically, we add $\mathcal{O}(d^n)$ terms together.

We consider all assignments $y = b$ that are X-consistent with the assignment $x = a$ within the whole problem to compute $\theta_x(a)$ for a given constraint.

4	3,5	1,5	1,3	2
1,3,5	3,5	1,4,5	2	1,3,4
1,3,5	2	1,4,5	1,3	1,3,4
2	1	3	4	5
1,3	4	2	5	1,3

4	3,5	1,5	1,3	2
1,3,5	3,5	1,4,5	2	1,3,4
1,3,5	2	1	1,3	1,3,4
2	1	3	4	5
1,3	4	2	5	1,3

$$\theta_{x_i}(v) = \frac{1}{\eta} \prod_{x_j \in \mathcal{X} \setminus x_i} \sum_{v' \in \hat{D}_{x_i=v}(x_j)} \theta_{x_j}(v')$$

where $\hat{D}_{x_i=v}(x_j)$ represents the reduced domain of the variable x_j after assigning $x_i = v$ and enforcing X-consistency on the CSP.

Complexity: $\mathcal{O}(n.d)$, i.e. $\mathcal{O}(n.\sqrt{n})$ for the QWH.

Advantages

- Better accuracy with an even stronger consistency
- Generic, thus easily implementable for every problem

Drawbacks

- Higher complexity, due to the propagation to the entire problem

Advantages

- Better accuracy with an even stronger consistency
- Generic, thus easily implementable for every problem

Drawbacks

- Higher complexity, due to the propagation to the entire problem

Outline

At every choice point, we:
- randomly initialize Θ,
- iteratively adjust Θ until convergence (or for a given number of iterations), and
- use the resulting Θ to select a variable-value pair.

But which one?

Intuitively, we pick the variable-value with the *highest bias*.

At every choice point, we:
- randomly initialize Θ,
- iteratively adjust Θ until convergence (or for a given number of iterations), and
- use the resulting Θ to select a variable-value pair.

But which one?

4	3,5	1,5	1,3	2
1,3,5	3,5	1,4,5	2	1,3,4
1,3,5	2	1,4,5	1,3	1,3,4
2	1	3	4	5
1,3	4	2	5	1,3

	Exact
$x_{11}=1$	0.00
$x_{11}=3$	0.20
$x_{11}=5$	0.80

Intuitively, we pick the variable-value with the *highest bias*.

At every choice point, we:
- randomly initialize Θ,
- iteratively adjust Θ until convergence (or for a given number of iterations), and
- use the resulting Θ to select a variable-value pair.

But which one?

4	3,5	1,5	1,3	2
1,3,5	3,5	1,4,5	2	1,3,4
1,3,5	2	1,4,5	1,3	1,3,4
2	1	3	4	5
1,3	4	2	5	1,3

	Exact	EMBPa	EMBP-Lsup	EMBP-Gsup
x_{11}=1	0.00	0.18	0.08	0.00
x_{11}=3	0.20	0.32	0.05	0.00
x_{11}=5	0.80	0.50	0.87	1.00

Intuitively, we pick the variable-value with the *highest bias*.

At every choice point, we:
- randomly initialize Θ,
- iteratively adjust Θ until convergence (or for a given number of iterations), and
- use the resulting Θ to select a variable-value pair.

But which one?

4	3,5	1,5	1,3	2
1,3,5	3,5	1,4,5	2	1,3,4
1,3,5	2	1,4,5	1,3	1,3,4
2	1	3	4	5
1,3	4	2	5	1,3

	Exact	EMBPa	EMBP-Lsup	EMBP-Gsup
$x_{11}=1$	0.00	0.18	0.08	0.00
$x_{11}=3$	0.20	0.32	0.05	0.00
$x_{11}=5$	0.80	0.50	0.87	1.00

Intuitively, we pick the variable-value with the *highest bias*.

At every choice point, we:
- randomly initialize Θ,
- iteratively adjust Θ until convergence (or for a given number of iterations), and
- use the resulting Θ to select a variable-value pair.

But which one?

4	3,5	1,5	1,3	2
1,3,5	3,5	1,4,5	2	1,3,4
1,3,5	2	1,4,5	1,3	1,3,4
2	1	3	4	5
1,3	4	2	5	1,3

	Exact	EMBPa	EMBP-Lsup	EMBP-Gsup
$x_{11}=1$	0.00	0.18	0.08	0.00
$x_{11}=3$	0.20	0.32	0.05	0.00
$x_{11}=5$	0.80	0.50 ★	0.87 ★	1.00 ★

Intuitively, we pick the variable-value with the *highest bias*.

We evaluate our methods on 3 benchmark problems and compare the results with the following heuristics:

rndMinDom		Randomly picks up a variable with the smallest domain size.
MaxSD	*(Zanarini et al.)*	Branches where it is likely to find a higher number of solutions.
IlogIBS	*(Refalo)*	Chooses first the variable whose instantiation triggers the largest search space reduction.
IlogAdvIBS	*(Refalo)*	Chooses a subset of 5 variables with the best approximated impacts.
$RSC - LA$	*(Correia et al.)*	Stands for Restricted Singleton Consistency Look-Ahead heuristic.
$RSC2 - LA$	*(Correia et al.)*	Maintains RSC for a subset of variables whose domain size equals 2.

Table: Computation time (in seconds), backtracks and percentage of solved instances for 40 hard QWH instances of order 30

heuristics	total time	avg btk	solved
rndMinDom	26328.2	1300056	56.8%
MaxSD	4939.1	3503	**100.0%**
IlogIBS	29017.0	1001570	45.0%
IlogAdvIBS	13795.8	914849	85.0%
RSC-LA	14019.9	856	95.0%
RSC2-LA	7178.7	4880	95.0%
EMBPa	13932.2	82158	79.0%
EMBP-Lsup	12814.2	6642	82.5%
EMBP-Gsup	**3946.9**	**55**	99.5%

Figure: Percentage of solved instances vs time for 40 hard QWH instances of order 30

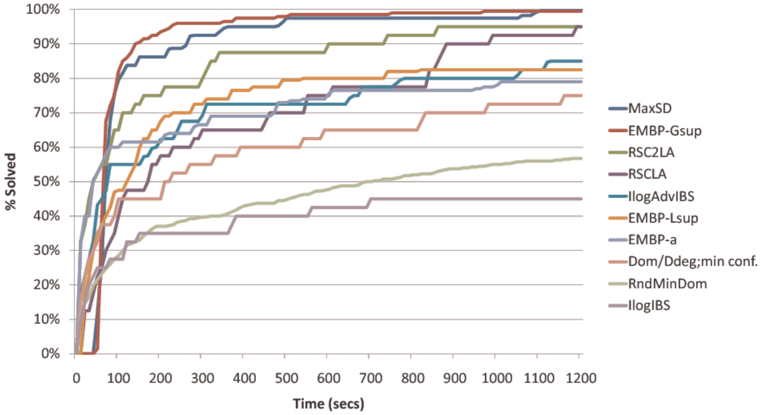


Table: Computation time (in seconds), number of backtracks and percentage of solved instances for 180 Nonogram instances

heuristics	total time	avg btk	solved
rndMinDom	20259.3	62662.4	83.8%
MaxSD	5029.6	8385	96.1%
IlogIBS	741.1	2665	99.4%
IlogAdvIBS	734.9	5866	99.4%
RSC-LA	580.9	10	**100.0%**
RSC2-LA	**432.5**	6	**100.0%**
EMBP-Lsup	5158.5	352	96.7%
EMBP-Gsup	813.9	**4**	99.4%

Figure: Percentage of solved instances vs time for 180 Nonogram instances

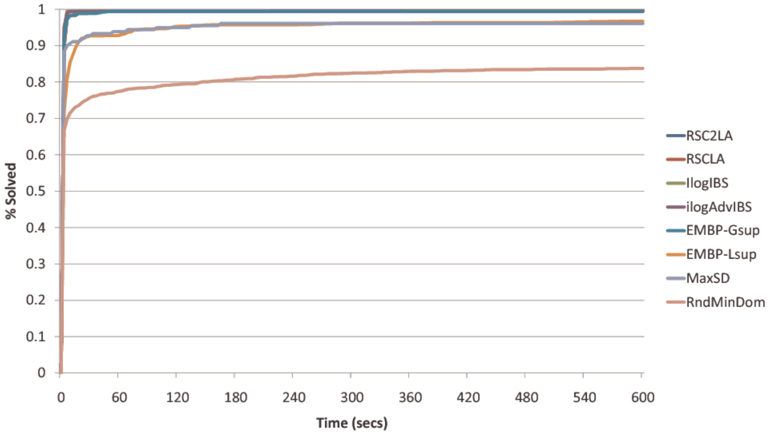
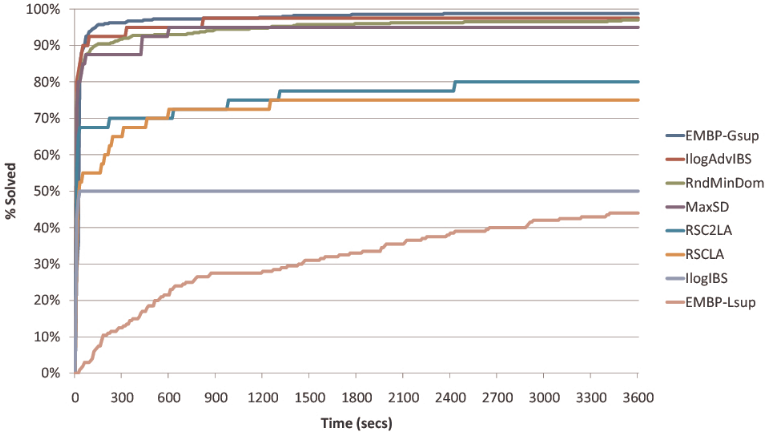


Table: Computation time (in seconds), number of backtracks and percentage of solved instances for 40 Magic Square instances

heuristics	total time	avg btk	solved
rndMinDom	7397.0	4018251	97.0%
MaxSD	8895.7	242290	95.0%
IlogIBS	72078.2	22396381	50.0%
IlogAdvIBS	5067.9	2224191	97.5%
RSC-LA	39612.4	48759	75.0%
RSC2-LA	34524.3	1180456	80.0%
EMBP-Lsup	98910.7	20572	43.0%
EMBP-Gsup	**3758.2**	**895**	**98.8%**

Figure: Percentage of solved instances vs time for 40 Magic Square instances



Summary

- We propose new efficient extensions to the EMBP framework.
- The methods are generic and can be applied to any CSP.
- EMBP-Gsup tends to be consistent and really competitive with existing approaches.

Future Work

- What is the most efficient way to use biases information? Possibilities are numerous to define a branching strategy based on the biases.
- How can we make EMBP-Gsup faster? Avoiding computing biases at every single node.
- Could we exploit *Solution Counting* to derive new EMBP update rules?

CORREIA, M., AND BARAHONA, P.
On the integration of singleton consistencies and look-ahead heuristics.
In *CSCLP* (2007), pp. 62–75.

HSU, E. I., KITCHING, M., BACCHUS, F., AND MCILRAITH, S. A.
Using expectation maximization to find likely assignments for solving csp's.
In *AAAI* (2007), pp. 224–230.

REFALO, P.
Impact-based search strategies for constraint programming.
In *CP* (2004), pp. 557–571.

ZANARINI, A., AND PESANT, G.
Solution counting algorithms for constraint-centered search heuristics.
Constraints 14, 3 (2009), 392–413.