Evaluation-Based Semiring Meta-Constraints

Jerome Kelleher and Barry O'Sullivan

4C, Department of Computer Science, UCC

jerome.kelleher@cs.ucc.ie

b.osullivan@cs.ucc.ie

This work has received support from Enterprise Ireland under their Basic Research Grant Scheme (Grant Number SC/02/289).

ENTERPRISE IRELAND

Summary

- Informal presentation of the semiring framework;
- Meta-constraints;
- Compilation vs Evaluation;
- Empirical Evaluation.

Semiring Framework - Informally

- Semiring framework provides an architecture for generalised problem solving.

- Specifies a class of problems using:
  - A set to represent all levels of consistency;
  - An operator which takes two elements of this set and returns the better (if either);
  - An operator which takes two elements of this set and returns the combination;
  - A lower bound on consistency; and,
  - An upper bound on problem consistency.

Soft Constraint Problems

To specify a particular problem within a class (semiring) we need:

- A set of variables;
- A set of constraints defined in terms of these variables;
  - ⋆ In the functional formulation constraints are *functions*[*Soft Concurrent Constraint Programming*, S. Bistarelli and U. Montanari and F. Rossi, ESOP '02].

Example Problem

- **Semiring:** $\langle \mathcal{R}^+, min, +, +\infty, 0 \rangle$;
- **Variables:** x, y and z, each defined over domain $\{1..4\}$;
- **Constraints:**
  - $\star$ $c_1\eta = x^{-\frac{2}{3}}y^{\frac{1}{2}}$
  - $\star$ $c_2\eta = z^{\frac{2}{3}}y^{-\frac{3}{2}}$

Meta-Constraints

- Constraints that involve other constraints;
- E.g. Combination, projection, solution, blevel;
- Many more possible.

ENTERPRISE IRELAND

Combination Meta-Constraints

- Allow us to treat a set of constraints as a single constraint;
- Ubiquitous operation in constraint processing algorithms;
- Very useful abstraction; shown here is pseudocode to find the value of a set of constraints C under an instantiation of the variables η, with and without the combination abstraction:

$$a \leftarrow \left(\bigotimes C\right)\eta$$

With combination

$a \leftarrow \mathbf{1}$
for all $c \in C$ **do**
  $a \leftarrow a \times c\eta$
end for

Without combination

Implementation by Compilation

x	y	$x^{-\frac{2}{3}}y^{\frac{1}{2}}$
1	1	1
1	2	1.414
1	3	1.732
1	4	2
2	1	0.63
2	2	0.891
2	3	1.091
2	4	1.26
3	1	0.481
3	2	0.68
3	3	0.833
3	4	0.961
4	1	0.397
4	2	0.561
4	3	0.687
4	4	0.794

c_1

$\otimes$

y	z	$z^{\frac{2}{3}}y^{-\frac{3}{2}}$
1	1	1
1	2	1.587
1	3	2.08
1	4	2.52
2	1	0.354
2	2	0.561
2	3	0.735
2	4	0.891
3	1	0.192
3	2	0.305
3	3	0.4
3	4	0.485
4	1	0.125
4	2	0.198
4	3	0.26
4	4	0.315

c_2

$=$

x	y	z	$x^{-\frac{2}{3}}y^{\frac{1}{2}} + z^{\frac{2}{3}}y^{-\frac{3}{2}}$
1	1	1	2
1	1	2	2.587
1	1	3	3.08
1	1	4	3.52
1	2	1	1.768
1	2	2	1.975
1	2	3	2.15
1	2	4	2.305
1	3	1	1.925
1	3	2	2.038
1	3	3	2.132
1	3	4	2.217
1	4	1	2.125
1	4	2	2.198
1	4	3	2.26
1	4	4	2.315
2	1	1	1.63
2	1	2	2.217
2	1	3	2.71
2	1	4	3.15
2	2	1	1.244
2	2	2	1.452
2	2	3	1.626
2	2	4	1.782
2	3	1	1.284
2	3	2	1.397
2	3	3	1.491
2	3	4	1.576
2	4	1	1.385
2	4	2	1.458
2	4	3	1.52
2	4	4	1.575
3	1	1	1.481
3	1	2	2.068
3	1	3	2.561
3	1	4	3.001
3	2	1	1.033
3	2	2	1.241
3	2	3	1.415
3	2	4	1.571
3	3	1	1.025
3	3	2	1.138
3	3	3	1.233
3	3	4	1.318
3	4	1	1.086
3	4	2	1.16
3	4	3	1.222
3	4	4	1.276
4	1	1	1.397
4	1	2	1.984
4	1	3	2.477
4	1	4	2.917
4	2	1	0.915
4	2	2	1.122
4	2	3	1.297
4	2	4	1.452
4	3	1	0.88
4	3	2	0.993
4	3	3	1.088
4	3	4	1.172
4	4	1	0.919
4	4	2	0.992
4	4	3	1.054
4	4	4	1.109

$c1 \otimes c2$

ENTERPRISE IRELAND

Implementation by Evaluation

x	y	$x^{-\frac{2}{3}}y^{\frac{1}{2}}$
1	1	1
1	2	1.414
1	3	1.732
1	4	2
2	1	0.63
2	2	0.891
2	3	1.091
2	4	1.26
3	1	0.481
3	2	0.68
3	3	0.833
3	4	0.961
4	1	0.397
4	2	0.561
4	3	0.687
4	4	0.794

c_1

$\otimes$

y	z	$z^{\frac{2}{3}}y^{-\frac{3}{2}}$
1	1	1
1	2	1.587
1	3	2.08
1	4	2.52
2	1	0.354
2	2	0.561
2	3	0.735
2	4	0.891
3	1	0.192
3	2	0.305
3	3	0.4
3	4	0.485
4	1	0.125
4	2	0.198
4	3	0.26
4	4	0.315

c_2

$$= \; c_1\eta \times_s c_2\eta$$

$c1 \otimes c2$

- Instead of *compiling* the combination constraint (exhaustively iterating through the cross product of the variables involved, computing the value of each instantiation when evaluated under each constraint, combining the values and storing the result), simply store the constraints involved and evaluate as required.

ENTERPRISE IRELAND

Lazy Evaluation

Algorithm 1 CombinationEvaluate(η)

$a \leftarrow$ **1**
for all $c \in C$ **do**
  $a \leftarrow a \times c\eta$
  if $a =$ **0 then**
    return 0
  end if
end for
return a

Compilation vs Evaluation

Criteria	*Compilation*	*Evaluation*
Time to allocate	Exponential	Linear
Space required	Exponential	Linear
Lookup Time	Constant	Linear

- Dynamic constraints very problematic for compilation - requires a complex update procedure at best; no effect on evaluation-based (as we store the *constraints*, not values);

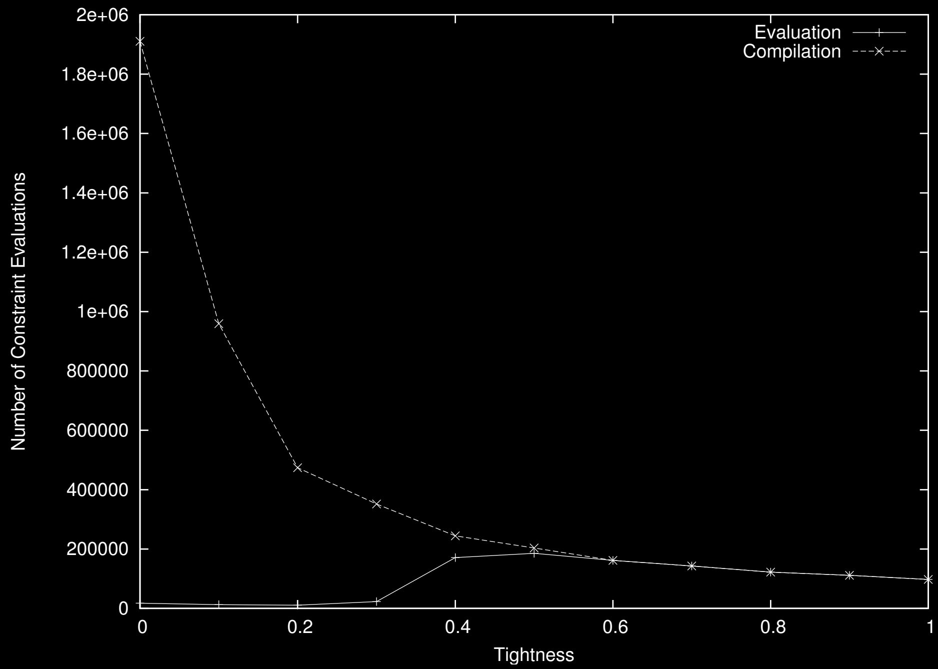
ENTERPRISE IRELAND

Empirical Evaluation

- Generated random Fuzzy (binary) CSPs with 7 variables, 5 domain value and with density 1.0. 50-fold cross validation of results;
- Count the number of constraint evaluations required to find the set of best solutions to these problems, i.e. the set of instantiations with the best semiring value when evaluated over the entire problem;
- Report results on problems with varying tightness.

ENTERPRISE IRELAND

Results



- Compilation *never* outperforms evaluation;
- Time decreases for compilation due to lazy evaluation.

ENTERPRISE
IRELAND

Conclusions

- Meta-constraints are a very useful abstraction;
- Any algorithm which utilises compilation-based meta-constraints will have exponential time *and* space complexity;
- In short, compilation of meta-constraints results in the computation and storage of a great deal of information which may not be necessary for a given task.