

Transformation-based Indexing Techniques for Constraint Handling Rules

Beata Sarna-Starosta Tom Schrijvers

Michigan State University & Logic Blox
BSS@CSE.MSU.EDU

Catholic University of Leuven, Belgium
TOM.SCHRIJVERS@CS.KULEUVEN.BE

CHR 2008 – July 14, 2008



- 1 Motivation
- 2 Generic Flattening
- 3 Symbol Specialization
- 4 Post-Processing
- 5 Evaluation
- 6 Conclusion



- 1 Motivation
- 2 Generic Flattening
- 3 Symbol Specialization
- 4 Post-Processing
- 5 Evaluation
- 6 Conclusion

A Historic Moment in CHR History (4 years ago)

4/40

Date: Thu, 15 Jul 2004 13:41:26 +0200

From: Thom Fruehwirth

To: CHR@LISTSERV.CC.KULEUVEN.AC.BE

Subject: [CHR] Programming Pearl: Merging and Sorting

It came to me last night that
a single rule merges and sorts:

$$X \sim > A \setminus X \sim > B \Leftrightarrow A @ < B \mid A \sim > B.$$

For details see attached file.

Thom

% merge two lists in order by zipping them together

$X \rightsquigarrow A \setminus X \rightsquigarrow B \iff \text{number}(X), A @ < B \mid A \rightsquigarrow B.$

% Go from quadratic to $n \log(n)$ complexity

% (assuming indexing) the number of elements

% N , must be a power of 2

$l(N) \rightsquigarrow A, l(N) \rightsquigarrow B \iff A @ < B \mid$

$M \text{ is } N+1, l(M) \rightsquigarrow A, A \rightsquigarrow B.$

Date: Wed, 21 Jul 2004 23:29:30 +0200

From: Tom Schrijvers

Subject: Re: [CHR] Programming Pearl: Merging and Sorting

Indeed seems to run in $n \cdot \log(n)$ time in hProlog
with \sim > separated out in two constraints
because K.U.L. CHR indexing is not working
with explicitly nested terms in the rules yet.

$X \sim> A \setminus X \sim> B \iff A@<B \mid A \sim> B.$

$\text{merge}(N,A), \text{merge}(N,B) \iff A@<B \mid$
 $M \text{ is } N+1, \text{merge}(M,A), A \sim> B.$

- ▶ CHR programmers should pay attention to how they write their programs?
- ▶ CHR indexing is no good?
 - ▶ attributed variables
 - ▶ search trees
 - ▶ hashtables

Not one system has one solution, . . .
. . . but two systems have two:

- ▶ Generic Flattening
- ▶ Symbol Specialization

in

- ▶ CHRd [Sarna-Starosta]
- ▶ K.U.Leuven CHR [Schrijvers et al.]



- 1 Motivation
- 2 Generic Flattening**
- 3 Symbol Specialization
- 4 Post-Processing
- 5 Evaluation
- 6 Conclusion

- ▶ flattening function

$$\phi(A) = \begin{cases} 1, N & , \text{ if } A = 1(N) \\ A, \epsilon & , \text{ otherwise} \end{cases}$$

- ▶ unflattening function

$$\psi(A, N) = \begin{cases} 1(N) & , \text{ if } A = 1 \\ A & , \text{ otherwise} \end{cases}$$

$$A \sim > B \iff \text{arrow}(\phi(A), B).$$

Flattened Program

12/40

```
% X ~> A \ X ~> B <=> number(X), A@<B | A ~> B.
```

```
arrow(F1,N1,A) \ arrow(F2,N2,B) <=>
  X =  $\psi$ (F1,N1), Y =  $\psi$ (F2,N2),
  number(X), A@<B, X = Y | A ~> B.
```

```
% l(N) ~> A, l(N) ~> B <=> A@<B |
%      M is N+1, l(M) ~> A, A ~> B.
```

```
arrow(l,N1,A), arrow(l,N2,B) <=>
  X =  $\psi$ (l,N1), Y =  $\psi$ (l,N2),
  A @< B, X = Y |
  M is N+1, l(M) ~> A, A ~> B.
```



- 1 Motivation
- 2 Generic Flattening
- 3 Symbol Specialization**
- 4 Post-Processing
- 5 Evaluation
- 6 Conclusion

- ▶ flattening function

$$\phi(A) = \begin{cases} N & , \text{ if } A = \mathbf{1}(N) \\ A & , \text{ otherwise} \end{cases}$$

- ▶ unflattening function

$$\begin{cases} \psi_I(N) & = \mathbf{1}(N) \\ \psi_\epsilon(A) & = A \end{cases}$$

$A \sim > B \iff A = 1(N)$
| $\text{merge}(\phi(A), B)$.

$A \sim > B \iff A \neq 1(N)$
| $\text{arrow}(\phi(A), B)$.

Flattened Program (1/2)

16/40

$\% X \rightsquigarrow A \ \backslash \ X \rightsquigarrow B \Leftrightarrow \text{number}(X), A@<B \mid A \rightsquigarrow B.$

$\text{arrow}(X1,A) \ \backslash \ \text{arrow}(X2,B) \Leftrightarrow$
 $X = \psi_\epsilon(X1), Y = \psi_\epsilon(X2),$
 $\text{number}(X), A@<B, X = Y \mid A \rightsquigarrow B.$

$\text{merge}(N1,A) \ \backslash \ \text{merge}(N2,B) \Leftrightarrow$
 $X = \psi_I(N1), Y = \psi_I(N2),$
 $\text{number}(X), A@<B, X = Y \mid A \rightsquigarrow B.$

Flattened Program (2/2)

17/40

```

% l(N) ~> A, l(N) ~> B <=> A@<B |
%      M is N+1, l(M) ~> A, A ~> B.
  
```

```

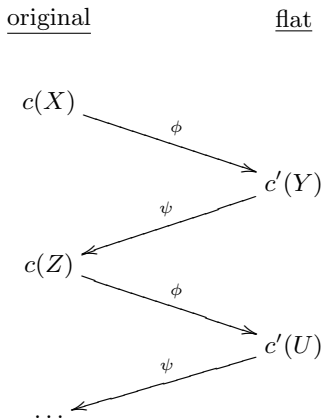
merge(N1,A), merge(N2,B) <=>
  X =  $\psi_l(N1)$ , Y =  $\psi_l(N2)$ ,
  A @< B, X = Y |
  M is N+1, l(M) ~> A, A ~> B.
  
```



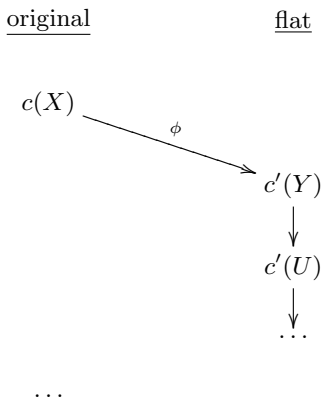
- 1 Motivation
- 2 Generic Flattening
- 3 Symbol Specialization
- 4 Post-Processing**
- 5 Evaluation
- 6 Conclusion

Conversion Overhead!

19/40



(a) Actual Situation



(b) Ideal Situation

Eliminating coverage overhead in 4 simple steps:

- 1 Make flattening explicit.
- 2 Eliminate flattening after unflattening.
- 3 Lift matching from terms to flattened terms.
- 4 Clean up.

Step 1: Make Flattening Explicit

21/40

$$\begin{aligned} \text{arrow}(l, N1, A), \text{ arrow}(l, N2, B) &\Leftrightarrow \\ X = \psi(l, N1), Y = \psi(l, N2), \\ A @< B, X = Y \mid \\ M \text{ is } N+1, \\ l(M) \sim> A, \\ A \sim> B. \end{aligned}$$

Step 1: Make Flattening Explicit

22/40

$$\begin{aligned} & \text{arrow}(l, N1, A), \text{ arrow}(l, N2, B) \Leftrightarrow \\ & X = \psi(l, N1), Y = \psi(l, N2), \\ & A @< B, X = Y \mid \\ & M \text{ is } N+1, \\ & \text{arrow}(\phi(l(M)), A), \\ & \text{arrow}(\phi(A), B). \end{aligned}$$

Step 1: Make Flattening Explicit

23/40

Allows partial evaluation of ϕ :

```
arrow(l,N1,A), arrow(l,N2,B) <=>
  X =  $\psi(l, N1)$ , Y =  $\psi(l, N2)$ ,
  A @< B, X = Y |
  M is N+1,
  arrow(l,M,A),
  arrow( $\phi(A)$ ,B).
```

$$\phi \circ \psi \equiv id$$

See paper. Not for this example.

Step 3: Lift Matching to Flattened Terms

25/40

$$\begin{aligned} \text{arrow}(F1, N1, A) \setminus \text{arrow}(F2, N2, B) \Leftrightarrow \\ X = \psi(F1, N1), Y = \psi(F2, N2), \\ \text{number}(X), A @ < B, X = Y \mid \\ \text{arrow}(\phi(A), B). \end{aligned}$$

Step 3: Lift Matching to Flattened Terms

26/40

$$\begin{aligned} \text{arrow}(F, N, A) \setminus \text{arrow}(F, N, B) &\Leftrightarrow \\ X = \psi(F1, N1), & \\ \text{number}(X), A @< B \mid & \\ \text{arrow}(\phi(A), B). & \end{aligned}$$

Refold remaining ϕ s:

$$\begin{aligned} \text{arrow}(F, N, A) \setminus \text{arrow}(F, N, B) &\Leftrightarrow \\ X = \psi(F1, N1), & \\ \text{number}(X), A @< B \mid & \\ \text{arrow}(\phi(A), B). & \end{aligned}$$

Step 4: Clean up

28/40

Refold remaining ϕ s:

$$\begin{aligned} \text{arrow}(F,N,A) \setminus \text{arrow}(F,N,B) &\Leftrightarrow \\ X = \psi(F1,N1), & \\ \text{number}(X), A @< B \mid & \\ A \sim > B. & \end{aligned}$$

Drop spurious ψ s:

```
arrow(1,N,A), arrow(1,N2B) <=>
  X =  $\psi(l, N1)$ ,
  A @< B |
  M is N+1,
  arrow(1,M,A),
  A ~> B.
```

Step 4: Clean up

30/40

Drop spurious ψ s:

```
arrow(1,N,A), arrow(1,N,B) <=>
  A @< B |
  M is N+1,
  arrow(1,M,A),
  A ~> B.
```

When the dust settles: Generic Flattening

31/40

$\% X \rightsquigarrow A \ \backslash \ X \rightsquigarrow B \iff \text{number}(X), A @< B \mid A \rightsquigarrow B.$

$\text{arrow}(F, N, A) \ \backslash \ \text{arrow}(F, N, B) \iff$
 $X = \psi(F, N),$
 $\text{number}(X), A @< B \mid A \rightsquigarrow B.$

$\% l(N) \rightsquigarrow A, l(N) \rightsquigarrow B \iff A @< B \mid$
 $\% \quad M \text{ is } N+1, l(M) \rightsquigarrow A, A \rightsquigarrow B.$

$\text{arrow}(l, N, A), \text{arrow}(l, N, B) \iff A @< B \mid$
 $M \text{ is } N+1, \text{arrow}(l, M, A), A \rightsquigarrow B.$

When the dust settles: Symbol Specialization

32/40

$\% X \rightsquigarrow A \ \backslash \ X \rightsquigarrow B \iff \text{number}(X), A @ < B \mid A \rightsquigarrow B.$

$\text{arrow}(X, A) \ \backslash \ \text{arrow}(X, B) \iff$
 $\text{number}(X), A @ < B \mid A \rightsquigarrow B.$

$\% l(N) \rightsquigarrow A, l(N) \rightsquigarrow B \iff A @ < B \mid$
 $\% \quad M \text{ is } N+1, l(M) \rightsquigarrow A, A \rightsquigarrow B.$

$\text{merge}(N, A), \text{merge}(l, N, B) \iff A @ < B \mid$
 $M \text{ is } N+1, \text{merge}(M, A), A \rightsquigarrow B.$



- 1 Motivation
- 2 Generic Flattening
- 3 Symbol Specialization
- 4 Post-Processing
- 5 Evaluation**
- 6 Conclusion

$O(n)$ vs. $O(1)$

```

check_birthdays(date(Day, Month, CurrentYear)),
  employee(Name, date(Day, Month, YearOfBirth)) ==>
    Age is CurrentYear - YearOfBirth,
  celebrate(Name, Age)
  
```

program version	number of employees		
	1,000	10,000	50,000
-flat -pp	2.000	22.000	108.000
+flat +pp	0.029	0.028	0.029

Evaluation: Generic Flattening

35/40

benchmark	K.U.Leuven CHR function symbols			CHRd function symbols		
	-flat	+flat	+flat	-flat	+flat	+flat
	-pp	-pp	+pp	-pp	-pp	+pp
gamma_prime	3.1	219%	137%	5.4	111%	93%
listdom	5.0	114%	108%	6.9	86%	84%
mergesort	1.9	592%	56%	6.6	113%	103%
ram_op	8.8	130%	96%	8.3	94%	89%
ram_prog	2.9	102%	96%	4.4	91%	86%
zebra	5.1	124%	93%	6.4	113%	102%

Evaluation: Symbol Specialization

36/40

benchmark	K.U.Leuven CHR function symbols			CHRd function symbols		
	-flat	+flat	+flat	-flat	+flat	+flat
	-pp	-pp	+pp	-pp	-pp	+pp
gamma_prime	1.5	114%	94%	4.6	93%	83%
listdom	5.2	174%	99%	7.2	129%	90%
manners	2.2	70%	65%	4.9	131%	124%
mergesort	7.8	30%	30%	6.7	82%	81%
ram_op	8.5	81%	82%	7.5	87%	88%
ram_prog	2.9	93%	93%	4.5	82%	80%
zebra	5.1	96%	91%	6.3	97%	97%
zebra2	4.8	103%	100%	6.9	96%	97%



- 1 Motivation
- 2 Generic Flattening
- 3 Symbol Specialization
- 4 Post-Processing
- 5 Evaluation
- 6 Conclusion**

Lessons learned:

- ▶ Programmers don't have to change their ways.
- ▶ CHR systems can deal with term patterns
- ▶ ... via generic flattening
- ▶ ... via symbol specialization
- ▶ ... and post-processing is essential.
- ▶ Program transformation for CHR works!

- ▶ Which approach to choose?
- ▶ Symbol specialization for (arbitrary) guards?
- ▶ Coming Soon: Attributed Data

Thank you!

40/40

Questions?

Further reading:

A Survey of CHR Research from 1998 to 2007.

J. Sneyers, P. Van Weert, T. Schrijvers, L. De Koninck.

<http://www.cs.kuleuven.be/~toms/>