Automatic **Inversion** Generates Generic **MapReduce** Parallel Programs

– Inversion in Program Construction –

Zhenjiang Hu
(Joint Work with A. Morihata, K. Matsuzaki, M. Takeichi)

GRACE Center
National Institute of Informatics

December 17, 2008

List Homomorphism (MapReduce)

Function h on lists is a list homomorphism, if

$$\begin{array}{lcl} h\ [a] & = & f\ \ a \\ h\ (x +\!\!+\, y) & = & h\ x \odot h\ y \end{array}$$

for some $\odot$.

Properties

- Suitable for parallel computation in the D&C style
- Basic concept for skeletal parallel programming
- Enjoy many nice algebraic properties (1st, 2nd, 3rd Homomorphism theorems)

An Example

- Maximum Prefix Sum Problem

Design a mapReduce parallel program that computes the maximum of all the prefix sums of a list.

$$mps\ [\underline{1, -2, 3, -9, 5, 7}, -10, 8, -9, 10] = 5$$

Parallelization Theorem

Let f° denote a weak right inverse of f.

$$\frac{\begin{array}{rcl} f([a] +\!\!+\, x) & = & a \oplus f\ x \\ f(x +\!\!+\, [b]) & = & f\ x \otimes b \end{array}}{\begin{array}{l} f(x +\!\!+\, y) = f\ x \odot f\ y \\ \quad \textbf{where}\ \ a \odot b = f(f^\circ\ a +\!\!+\, f^\circ\ b) \end{array}}$$

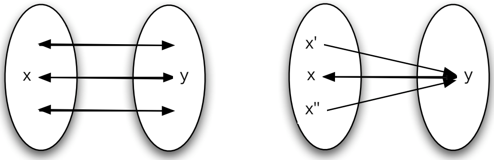
Weak (Right) Inverse

- g is an inverse of f, if

$$g\ y = x \ \Leftrightarrow\ f\ x = y$$

- g is a weak (right) inverse of f, if for $y \in \mathrm{image}(f)$

$$g\ y = x \ \Rightarrow\ f\ x = y$$



Properties of Weak Inverse

- Weak inverse always exists but may not be unique.

  Example: Function *sum*

$$\begin{aligned} sum\ [\,] &= 0 \\ sum\ (a : x) &= a + sum\ x \end{aligned}$$

  can have infinite number of weak inverse:

Properties of Weak Inverse

- Weak inverse always exists but may not be unique.

  Example: Function *sum*

$$\begin{array}{lcl} sum\ [\,] & = & 0 \\ sum\ (a : x) & = & a + sum\ x \end{array}$$

  can have infinite number of weak inverse:

$$\begin{array}{lcl} g_1\ y & = & [y] \\ g_2\ y & = & [0, y] \\ & \cdots & \end{array}$$

Parallelizing *sum*

From

1. $sum\ (a : x) = a + sum\ x$
2. $sum\ (x +\!\!+ [b]) = sum\ x + b$
3. $sum^\circ\ y = [y]$

we soon obtain

$$
\begin{array}{l}
sum\ (x +\!\!+ y) = sum\ x \odot sum\ y \\
\quad \text{where} \\
\qquad \begin{array}{rcl}
a \odot b & = & sum\ (sum^\circ\ a +\!\!+ sum^\circ\ b) \\
& = & sum\ ([a] +\!\!+ [b]) \\
& = & a + b
\end{array}
\end{array}
$$

Parallelizing *sum*

From

1. $sum\ (a : x) = a + sum\ x$
2. $sum\ (x +\!\!+ [b]) = sum\ x + b$
3. $sum^{\circ}\ y = [y]$

we soon obtain

$$
\begin{array}{l}
sum\ (x +\!\!+ y) = sum\ x \odot sum\ y \\
\quad \text{where} \\
\qquad \begin{array}{rcl}
a \odot b & = & sum\ (sum^{\circ}\ a +\!\!+ sum^{\circ}\ b) \\
& = & sum\ ([a] +\!\!+ [b]) \\
& = & a + b
\end{array}
\end{array}
$$

That is,

$$sum\ (x +\!\!+ y) = sum\ x + sum\ y.$$

Weak inversion is not easy!

- What is a weak inverse for *sum*?

$$\begin{array}{lcl} sum\ [\,] & = & 0 \\ sum\ (a : x) & = & a + sum\ x \end{array}$$

Weak inversion is not easy!

- What is a weak inverse for *sum*? $\underline{sum^{\circ}\ y = [y]}$

$$
\begin{array}{lcl}
sum\ [\,] & = & 0 \\
sum\ (a : x) & = & a + sum\ x
\end{array}
$$

Weak inversion is not easy!

- What is a weak inverse for *sum*? $\underline{sum^{\circ}\ y = [y]}$

$$\begin{array}{lcl} sum\ [\,] & = & 0 \\ sum\ (a : x) & = & a + sum\ x \end{array}$$

- What is it for *mps*?

$$\begin{array}{l} mps\ [\,] = 0 \\ mps\ (a : x) = 0 \uparrow a \uparrow (a + mps\ x) \end{array}$$

Weak inversion is not easy!

- What is a weak inverse for *sum*? $\underline{sum^\circ\ y = [y]}$

$$\begin{array}{lcl} sum\ [\,] & = & 0 \\ sum\ (a:x) & = & a + sum\ x \end{array}$$

- What is it for *mps*? $\underline{mps^\circ\ y = [y]}$

$$\begin{array}{l} mps\ [\,] = 0 \\ mps\ (a:x) = 0 \uparrow a \uparrow (a + mps\ x) \end{array}$$

Weak inversion is not easy!

- What is a weak inverse for *sum*? $\underline{sum^\circ\ y = [y]}$

$$\begin{array}{lcl} sum\ [\,] & = & 0 \\ sum\ (a : x) & = & a + sum\ x \end{array}$$

- What is it for *mps*? $\underline{mps^\circ\ y = [y]}$

$$\begin{array}{l} mps\ [\,] = 0 \\ mps\ (a : x) = 0 \uparrow a \uparrow (a + mps\ x) \end{array}$$

- What is it for $f = mps \vartriangle sum$?

$$f\ \ x = (mps\ x, sum\ x)$$

Weak inversion is not easy!

- What is a weak inverse for *sum*? $\underline{sum^\circ\ y = [y]}$

$$\begin{array}{lcl} sum\ [\,] & = & 0 \\ sum\ (a : x) & = & a + sum\ x \end{array}$$

- What is it for *mps*? $\underline{mps^\circ\ y = [y]}$

$$\begin{array}{l} mps\ [\,] = 0 \\ mps\ (a : x) = 0 \uparrow a \uparrow (a + mps\ x) \end{array}$$

- What is it for $f = mps \vartriangle sum$? $\underline{f^\circ\ (p, s) = [p, s - p]}$

$$f\ \ x = (mps\ x, sum\ x)$$

Weak inversion is challenging

Can you find a weak inverse for f?

$$f\ x = (mss\ x, mps\ x, mts\ x, sum\ x)$$

where

$$\begin{array}{lcl} mss\ [\,] & = & 0 \\ mss\ (a : x) & = & (a + mps\ x) \uparrow mss\ x \uparrow 0 \\ mts\ [\,] & = & 0 \\ mts\ (a : x) & = & (a + sum\ x) \uparrow mts\ x \uparrow 0 \end{array}$$

Weak inversion is challenging

Can you find a weak inverse for f?

$$f\ x = (mss\ x, mps\ x, mts\ x, sum\ x)$$

where

$$\begin{array}{lcl} mss\ [\,] & = & 0 \\ mss\ (a : x) & = & (a + mps\ x) \uparrow mss\ x \uparrow 0 \\ mts\ [\,] & = & 0 \\ mts\ (a : x) & = & (a + sum\ x) \uparrow mts\ x \uparrow 0 \end{array}$$

$$\underline{f^\circ\ (m, p, t, s) = [p, s - p - t, m, t - m]}$$

Derivation of Weak Right Inverse

- Idea:

deriving a weak right inverse

$\Downarrow$

solving conditional linear equations

Derivation of Weak Right Inverse

- Idea:

deriving a weak right inverse
$\Downarrow$
solving conditional linear equations

- Consider to find a weak right inverse for f defined by

$$f\ x = (mps\ x, sum\ x)$$

Let x_1, x_2 be a solution to the following equations:

$$\begin{array}{rcl} mps\ [x_1, x_2] & = & p \\ sum\ [x_1, x_2] & = & s \end{array}$$

then

$$f^{\circ}\ (p, s) = [x_1, x_2]$$

Derivation of Weak Right Inverse

- Idea:

  deriving a weak right inverse
  $$\Downarrow$$
  solving conditional linear equations

- Consider to find a weak right inverse for f defined by

$$f\ x = (mps\ x, sum\ x)$$

Let x_1, x_2 be a solution to the following equations:

$$\begin{array}{lcl} 0 \uparrow x_1 \uparrow (x_1 + x_2) & = & p \\ x_1 + x_2 & = & s \end{array}$$

then

$$f^\circ\ (p, s) = [x_1, x_2]$$

Derivation of Weak Right Inverse

- Idea:

  deriving a weak right inverse

  $\Downarrow$

  solving conditional linear equations

- Consider to find a weak right inverse for f defined by

$$f\ x = (mps\ x, sum\ x)$$

Let x_1, x_2 be a solution to the following equations:

$$\begin{aligned} x_1 &= p \\ x_2 &= s - p \end{aligned}$$

then

$$f^{\circ}\ (p, s) = [x_1, x_2]$$

Derivation of Weak Right Inverse

- Idea:

  deriving a weak right inverse

  $\Downarrow$

  solving conditional linear equations

- Consider to find a weak right inverse for f defined by

$$f\ x = (mps\ x, sum\ x)$$

Let x_1, x_2 be a solution to the following equations:

$$\begin{array}{rcl} x_1 & = & p \\ x_2 & = & s - p \end{array}$$

then

$$f^\circ\ (p, s) = [p, s - p]$$

Conditional Linear Equations

$$
\begin{aligned}
t_1(x_1, x_2, \ldots, x_m) &= c_1 \\
t_2(x_1, x_2, \ldots, x_m) &= c_2 \\
&\vdots \\
t_m(x_1, x_2, \ldots, x_m) &= c_m
\end{aligned}
$$

Conditional Linear Equations

$$
\begin{array}{rcl}
t_1(x_1, x_2, \ldots, x_m) & = & c_1 \\
t_2(x_1, x_2, \ldots, x_m) & = & c_2 \\
 & \vdots & \\
t_m(x_1, x_2, \ldots, x_m) & = & c_m
\end{array}
$$

$$
\begin{array}{rcl}
t & ::= & n \mid x \mid n\ x \mid t_1 + t_2 \mid p \rightarrow t_1; t_2 \\
p & ::= & t_1 < t_2 \mid t_1 = t_2 \mid \neg p \mid p_1 \wedge p_2 \mid p_1 \vee p_2
\end{array}
$$

Conditional Linear Equations

$$\begin{array}{rcl} t_1(x_1, x_2, \ldots, x_m) & = & c_1 \\ t_2(x_1, x_2, \ldots, x_m) & = & c_2 \\ & \vdots & \\ t_m(x_1, x_2, \ldots, x_m) & = & c_m \end{array}$$

Conditional linear equations can be efficiently solved by using Mathematica. [PLDI'07]

Can we generalize the idea from lists to trees?

$$\frac{\begin{array}{lcl} f(a:x) & = & a \oplus f\ x \\ f(x +\!\!+ [b]) & = & f\ x \otimes b \end{array}}{\begin{array}{l} f(x +\!\!+ y) = f\ x \odot f\ y \\ \quad \textbf{where}\ \ a \odot b = f(f^{\circ}\ a +\!\!+ f^{\circ}\ b) \end{array}}$$

$$\Downarrow$$

$$\frac{\begin{array}{l} \text{f is a bottom-up tree reduction} \\ \text{f is a top-down tree reduction} \end{array}}{\begin{array}{l} f(t_1 \triangleleft t_2) = f\ t_1 \odot f\ t_2 \\ \quad \textbf{where}\ \ a \odot b = f(f^{\circ}\ a \triangleleft f^{\circ}\ b) \end{array}}$$

Can we generalize the idea from lists to trees?

$$\frac{\begin{array}{lcl} f(a : x) & = & a \oplus f\ x \\ f(x +\!\!+ [b]) & = & f\ x \otimes b \end{array}}{\begin{array}{l} f(x +\!\!+ y) = f\ x \odot f\ y \\ \quad \textbf{where}\ \ a \odot b = f(f^\circ\ a +\!\!+ f^\circ\ b) \end{array}}$$

$$\Downarrow$$

$$\frac{\begin{array}{l} \text{f is a bottom-up tree reduction} \\ \text{f is a top-down tree reduction} \end{array}}{\begin{array}{l} f(t_1 \triangleleft t_2) = f\ t_1 \odot f\ t_2 \\ \quad \textbf{where}\ \ a \odot b = f(f^\circ\ a \triangleleft f^\circ\ b) \end{array}}$$

To be presented in POPL'09.

Concluding Remarks

- Right inverse is a create function in bidirectional computation.
- Right inversion is helpful for automatic program construction.

References

1. Basic Idea
   Automatic Inversion Generates Divide-and-Conquer Parallel Programs, *ACM SIGPLAN 2007 Conference on Programming Language Design and Implementation* (PLDI'07), San Diego, CA, USA, June 10-13, 2007.

2. A Generalization
   The Third Homomorphism Theorem on Trees: Downward & Upward Lead to Divide-and-Conquer, *36th Annual ACM SIGPLAN - SIGACT Symposium on Principles of Programming Languages* (POPL 2009), Savannah, Georgia, USA, January 21-23, 2009.