Using Speculative Computation and Parallelizing techniques to improve Scheduling of Control based Designs

Roberto Cordone             Fabrizio Ferrandi, Gianluca Palermo, Marco D. Santambrogio, Donatella Sciuto

Università Statale di Milano - DTI         Politecnico di Milano - DEI

ASP-DAC 2006 Yokohama, Japan 24-27 January 2006
Outline

- Previous approaches on scheduling with speculation
- Analysis of IRs for HLS:
  - PDG/SDG
  - CDFG/HTG
- Scheduling:
  - Speculation
  - Transformation
- Scheduling ILP formulation:
  - new conditional resource sharing constraint
  - Speculative computation
- Experimental results
Previous approaches

- Code motion techniques
  - SW compilers: Fisher81, Nicolau89
  - HLS: Santos99, Rim95
- Speculation: Jha99, Gupta03, Brewer96
- No exact methods with the exception of Brewer96
  - CDFG, Unit time, no pipeline unit
- Previous ILP-based scheduling (Gebotys93): does not well express control constraints
- Intermediate Representation: CDFG, HTG
What PDGs and SDGs are

- PDGs are the starting point: they represent a single procedure
- A PDG is a directed graph
  - Its nodes represent:
    - Statements
    - Predicates (loop/control conditions)
  - Its edges represent:
    - Data dependencies
    - Control dependencies
- A System Dependency Graph SDG is a collection of PDGs connected by call and parameter edges
- System Dependency Graphs:
  - Abstract code representation
  - Explicit representation of all dependencies between statements
  - Easy detection of parallelizable code
Control dependencies

- **Intuition:**
  - Node $A$ is control dependent on node $B$ if $B$ may change whether $A$ is executed or not

- **Formal definition (Ferrante et al.):**
  - $Y$ is control dependent on $X$ iff:
    - There exists a path $P$ from $X$ to $Y$ in the CFG with any node $Z$ in $P$ post-dominated by $Y$
    - $X$ is not post-dominated by $Y$
CFG vs CDG (Ferrante + Girkar & Polychronopoulos)
void
gcd (int xi, int yi, int *ou)
{
    int x, y, temp;
    x = xi;
    y = yi;
    while (x >0)
    {
        if (x <= y)
        {
            temp = x;
            x = y-x;
            y = temp;
        }
        else
        {
            x = x - y;
        }
    }
    *ou = y;
}

Transformations vs scheduling
Gupta et al.

- **Across Hierarchical Blocks**
  - movement of operations across entire hierarchical blocks

- **Speculation**
  - unconditional execution of operations that were originally supposed to have executed conditionally

- **Reverse Speculation**
  - where operations before conditionals are moved into subsequent conditional blocks and executed conditionally

- **Conditional Speculation**
  - in which an operation is moved and duplicated up into preceding conditional branches and executed conditionally
Across Hierarchical Blocks

HTG_0

HTG_1

BB0
cond = a < b
b = c - 1

BB1

BB2
x = a + b

BB3
a = c - d
x = a · b

BB4

BB5
y = e + f
z = y + x

ENTRY

EXIT

b = e - f
cond = a < b
y = e + f

IF0<cond>

b
cond
T
F

x = a + b

x = a - b

a = c - d

y

x

z = y + x
Speculation & Conditional Speculation
Reverse Speculation

HTG₀

BB₀
\(c = in₁ < in₂\)
\(b = in₁ + in₂\)

BB₁

BB₂
\(d = in₁ + in₃\)
\(e = in₁ + d\)
\(x = in₃ + e\)

BB₃
\(x = in₃ + b\)

BB₄

BB₅
\(ou = x + in₂\)

ENTRY

EXIT

\(c = in₁ < in₂\)
\(b = in₁ + in₂\)

IF0\(<c>\)

\(x = in₃ + b\)
\(d = in₁ + in₃\)
\(c = in₁ + d\)
\(e = x + in₂\)
\(ou = x + in₂\)
Scheduling: ILP formulation (Gebotys)

Considered scheduling problem

\[ \begin{align*}
\min & (w) \\
\text{subject to} & \\
  w & \geq \sum_{i,j} (j+C_{k,i}-1)x_{i,j,k} \quad j \in J, \ k \in K
\end{align*} \]

where

- \( w \) is a variable representing the last control step
- \( x_{i,j,k} = 1 \) when operation \( k \) starts executing at control step \( j \) and it is assigned to a functional unit of type \( i \) and
- \( C_{k,i} \) is the execution time of operation \( k \) mapped on functional unit \( i \)
- \( L_{i,k} \) is the initiation time of operation \( k \) mapped on functional unit \( i \)
Scheduling constraints (Gebotys)

- **Assignment constraint**
  - Each operation is assigned to a specific control step
    \[ \sum_i \sum_j x_{i,j,k} = 1 \quad \forall k \in K \]

- **Precedence constraint**
  \[
  \sum_i \left( \sum_{j' \leq j_c, \text{asap}(k') \leq j' \leq \text{alap}(k')} x_{i,j',k'} + \sum_{j \geq j_c - C_{k,i} + 1, \text{asap}(k) \leq j \leq \text{alap}(k)} x_{i,j,k} \right) \leq 1
  \]
  Node packing problem

\( \forall k \in K, k' \in K, j_c \in J \) subject to
\( k \triangleleft k', k \notin N, k' \notin W \) and
\( \max(\text{asap}(k'), \text{asap}(k) + \min_i(C_k, i) - 1) \leq j_c \leq \min(\text{alap}(k'), \text{alap}(k) + \max_i(C_k, i) - 1) \)
Resource constraints

- **Conditional branch**

- **Gebotys et al.**
  - for each path defined a capacity constraints

- **Our approach**
  - recursive capacity constraints

\[ z_{i,j,B_0} \leq N_i \quad \forall i \in I, j \in J \]

\[ \sum_{k \in O_p} \sum_{j' = j - L_{j,k} + 1}^j x_{i,j',k} + \sum_{B \in B_p} z_{i,j,B} \leq z_{i,j,B} \]

\[ \forall i \in I, j \in J, B \in \beta, P \in B \]
To take into account speculation we modify the resource constraint:

\[
\sum_{P \in B} \left( \sum_{k \in P} \sum_{j' = j}^{j - L_{ki} + 1} x_{ij'k} + \sum_{B' \in P} z_{ijB'} \right) \leq z_{ijB} + M \left( 1 - \sum_{i' \in I_B} \sum_{j' \geq j - C_{kB} - 1} \sum_{B' \in P} x_{ij'B} \right)
\]

\(i \in I, j \in J, B \in B \setminus \{B_0\}\)

- \(M\) is a constant large enough to make the constraint redundant when no speculation is performed
SDG & Resource constraint with speculation
Branch and Cut

- Based on the open source package COIN-OR
  - (http://www.coin-or.org)
- provides a set of tools among which an ILP solver with the capability of generating the most important families of valid inequalities.

- The inequalities effective to solve the scheduling problem are:
  - Gomory
  - Clique
  - Probing
  - Knapsack
Experimental Results

- compare three different scheduling techniques
  - SPARK: Gupta et al. Framework
  - LIST: standard list based adapted to SDG
  - ILP: our ILP formulation of the scheduling problem with speculation

- Benchmarks:
  - A set of standard HLS benchmarks
  - Two media benchmarks:
    - MotionVector
    - Adpcm(Decode/Encode)
Experimental Results: speculation for ARCH1

<table>
<thead>
<tr>
<th>1-Add, 1-Sub, 1-Mul, 1-Cmp, 1-Sh, 2[]</th>
<th>SPARK</th>
<th>LIST</th>
<th>ILP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CS</td>
<td>Time(s)</td>
<td>CS</td>
</tr>
<tr>
<td>Kim</td>
<td>10</td>
<td>0.030</td>
<td>10</td>
</tr>
<tr>
<td>Sehwa</td>
<td>8</td>
<td>0.046</td>
<td>9</td>
</tr>
<tr>
<td>Maha</td>
<td>9</td>
<td>0.049</td>
<td>9</td>
</tr>
<tr>
<td>MotionVector</td>
<td>15</td>
<td>0.309</td>
<td>12</td>
</tr>
<tr>
<td>AdpcmDecode</td>
<td>18</td>
<td>0.110</td>
<td>13</td>
</tr>
<tr>
<td>AdpcmEncode</td>
<td>18</td>
<td>0.144</td>
<td>14</td>
</tr>
</tbody>
</table>
### Experimental Results: speculation for ARCH2

<table>
<thead>
<tr>
<th>1-Add, 1-Sub, 1-Mul, 2-Cmp, 1-Sh, 2[]</th>
<th>SPARK</th>
<th>LIST</th>
<th>ILP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CS</td>
<td>Time(s)</td>
<td>CS</td>
</tr>
<tr>
<td>Kim</td>
<td>10</td>
<td>0.054</td>
<td>10</td>
</tr>
<tr>
<td>Sehwa</td>
<td>7</td>
<td>0.045</td>
<td>7</td>
</tr>
<tr>
<td>Maha</td>
<td>9</td>
<td>0.054</td>
<td>9</td>
</tr>
<tr>
<td>MotionVector</td>
<td>13</td>
<td>0.274</td>
<td>12</td>
</tr>
<tr>
<td>AdpcmDecode</td>
<td>15</td>
<td>0.122</td>
<td>11</td>
</tr>
<tr>
<td>AdpcmEncode</td>
<td>18</td>
<td>0.145</td>
<td>13</td>
</tr>
</tbody>
</table>
### Experimental Results: speculation for ARCH3

<table>
<thead>
<tr>
<th>2-Add, 2-Sub, 1-Mul, 2-Cmp, 1-Sh, 2[]</th>
<th>SPARK</th>
<th>LIST</th>
<th>ILP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CS</td>
<td>Time(s)</td>
<td>CS</td>
</tr>
<tr>
<td>Kim</td>
<td>8</td>
<td>0.036</td>
<td>8</td>
</tr>
<tr>
<td>Sehwa</td>
<td>6</td>
<td>0.049</td>
<td>6</td>
</tr>
<tr>
<td>Maha</td>
<td>9</td>
<td>0.054</td>
<td>9</td>
</tr>
<tr>
<td>MotionVector</td>
<td>10</td>
<td>0.323</td>
<td>10</td>
</tr>
<tr>
<td>AdpcmDecode</td>
<td>15</td>
<td>0.114</td>
<td>10</td>
</tr>
<tr>
<td>AdpcmEncode</td>
<td>18</td>
<td>0.148</td>
<td>13</td>
</tr>
</tbody>
</table>
Experimental Results: B&B vs B&C

- Set of benchmarks enriched with some well known data-intensive high level synthesis benchmarks
- Two architecture considered: ARCH1 and ARCH3
- ILP branch & Bound:
  - Solution proved optimally in less than 1000sec: 15 vs 11
- ILP branch & Cut:
  - Solution proved optimally in less than 1000sec: 24 vs 2
- Further COIN-OR customization allow to optimally solve all the problems
Future works

- Improve the GCC interface
- Better support of reverse and conditionally speculation techniques
- Analysis of heuristics and inequalities to better support B&C: approximations, lower bound estimations
- Coin-Or branching customization
- Analysis and integration of register binding, module and interconnect allocation
- Exploitation of SDG to perform partitioning for
  - HW/SW Codesign
  - Dynamic reconfiguration
Any Questions?

ferrandi@elet.polimi.it