Optimal Placement of Bank Selection Instructions in Polynomial Time

Philipp Klaus Krause

2013-06-19
Table of Contents

1  Bank Selection

2  Tree Decompositions

3  Algorithm

4  Example

5  Summary
Partitioned Memory Architectures

- Allow using a physical address space larger than the logical one
- Common in embedded systems
- Bank selection instructions change which part of the physical address space is mapped into a window in logical address space
Figure 24. Physical Address Transition

The MMU block diagram is depicted in Figure 25. The MMU translates internal 16-bit logical addresses to external 20-bit physical addresses.

Figure 25. MMU Block Diagram

Logical Address Space

Common Area 1
Bank Area
Common Area 0

Physical Address Space

Common Base
Bank Base

FFFFFH
0000H

x y z

Logical Address Space

0

Physical Address Space

FFFFFH
00000H

x y z

Common Area 1
Bank Area
Common Area 0

z

y

x

Zilog Z180 manual
Placement of Bank Selection Instructions

- Inserting bank selection instructions optimally is hard
- Previous approaches are not optimal or do not have polynomial runtime
- Graph structure theory allows an efficient, optimal approach
int binomial_coefficient(int n, int k)
{
    int i, delta, max, c;
    if(n < k)
        return(0);
    if(n == k)
        return(1);
    if(k < n - k)
    {
        delta = n - k;
        max = k;
    }
    else
    {
        delta = k;
        max = n - k;
    }
    c = delta + 1;
    for(i = 2; i <= max; i++)
        c = (c * (delta + i)) / i;
    return(c);
}
```c
int binomial_coefficient(int n, int k)
{
    int i, delta, max, c;
    if(n < k)
        return(0);
    if(n == k)
        return(1);
    if(k < n - k)
    {
        delta = n - k;
        max = k;
    }
    else
    {
        delta = k;
        max = n - k;
    }
    c = delta + 1;
    for(i = 2; i <= max; i++)
        c = (c * (delta + i)) / i;
    return(c);
}
```
Given a graph $G$ with nodes $\Pi$ a \textit{tree decomposition} of $G$ is a pair $(T, \mathcal{X})$ of a tree $T$ and a family $\mathcal{X} = \{X_i \mid i \text{ node of } T\}$ of subsets of $\Pi$:

- $\bigcup_{i \text{ node of } T} X_i = \Pi$,
- $\forall\{x, y\} \text{ edge in } \Pi: \exists i \text{ node of } T: x, y \in X_i$,
- $\forall x \in \Pi: \{i \text{ node of } T \mid x \in X_i\}$ is connected in $T$. 

[Diagram of a tree decomposition with nodes labeled a, b, c, d, e, f, g, h, i, j, k and subsets $X_i$ indicated as circles containing subsets of nodes.]
Tree Width

- The width of a tree decomposition \((T, \mathcal{X})\) is
  \[\max\{|X_i| \mid \text{i node of } T\} - 1.\]
- The tree-width \(tw(G)\) of a graph \(G\) is the minimum width over all tree decompositions of \(G\).
- The tree-width of control-flow graphs is bounded\(^1\).

---

\(^1\)M. Thorup, 1998
A tree decomposition \((T, \mathcal{X})\) of a graph \(G\) is called nice, iff

- \(T\) is oriented, with root \(t, X_t = \emptyset\).
- Each node \(i\) of \(T\) is of one of the following types:
  - Leaf node, no children
  - Introduce node, one child \(j, X_j \subsetneq X_i\)
  - Forget node, one child \(j, X_j \supsetneq X_i\)
  - Join node, two children \(j_1, j_2, X_i = X_{j_1} = X_{j_2}\)
Table of Contents

1   Bank Selection
2   Tree Decompositions
3   Algorithm
4   Example
5   Summary
Notation

- Let \( \mathcal{B} \) be the set of memory banks in the architecture.
- Let \( G = (\Pi, E) \) be the control-flow graph of the program.
- Let \( c : E \times \mathcal{B} \times \mathcal{B} \to \mathbb{R} \) be the cost function.
- Let \( (T, \mathcal{X}) \) be a nice tree decomposition of minimum width of \( G \) with root \( t \).
For a fixed set of memory banks $\mathcal{B}$, the problem of Optimal Placement of Bank Selection Instructions with input program consisting of a partially colored control-flow graph $G = (\Pi, E)$ and cost function $c$ is the following: Find a compatible coloring $f : \Pi \rightarrow \mathcal{B}$ of $G$, such that

$$\sum_{e \in E} c(e, f(e_0), f(e_1))$$

is minimal over all compatible colorings.
$s(i, f)$ gives the minimum possible cost for the edges covered in
the subtree rooted at $i$, excluding the edges between nodes in in
$X_i$, when using $f$ as the coloring for $X_i$.

- **Leaf**: $s(i, f) := 0$.
- **Introduce**: $s(i, f) := \begin{cases} \infty & \text{if coloring incompatible} \\ s(j, f|_{X_j}) & \text{otherwise.} \end{cases}$
- **Forget**: $s(i, f) := \min \left\{ s(j, g) + \sum_{e \in (E \cap (X^2_j \setminus X^2_i))} c(e, g(e_0), g(e_1)) \mid g|_{X_i} = f \right\}$.  

- **Join**: $s(i, f) := s(j_1, f) + s(j_2, f)$.  

The optimal coloring in polynomial time

- The nice tree-decomposition of minimum width can be calculated in linear time\(^2\).
- \(s\) can be calculated in time \(O(|T| \cdot \text{tw}(G)^2 |\mathcal{B}|^{\text{tw}(G)+1})\).\(^2\)
- Extend the single remaining \(f\) at the root \(t\) to all nodes to obtain the optimal coloring.

\(^2\)H. Bodlaender 1996
Table of Contents

1 Bank Selection
2 Tree Decompositions
3 Algorithm
4 Example
5 Summary
if (...) 
    access_bank_a;
else 
    do_not_access_banked_memory;

switch (...) 
{
    case 0:
        do_not_access_banked_memory;
        break;
    case 1:
        access_bank_b; break;
    case 2:
        do_not_access_banked_memory;
        break;
    .
    .
    .
    case n-1:
        access_bank_b;
        break;
    default:
        do_not_access_banked_memory;
        break;
} 

access_bank_a;
Table of Contents

1  Bank Selection

2  Tree Decompositions

3  Algorithm

4  Example

5  Summary
Summary

- Optimal placement of bank selection instructions in polynomial time
- Linear time for fixed number of banks
- Previous approaches were not optimal or had no polynomial runtime bounds
- Uses concepts from graph-structure theory (tree-decompositions)
- Implemented in sdcc