Polyhedral Compilation Foundations

Louis-Noël Pouchet
pouchet@cse.ohio-state.edu

Dept. of Computer Science and Engineering, the Ohio State University

Feb 15, 2010

888.11, Class #4
Overview of Today’s Lecture

Outline:
- Solution of the exercise
- Linear Programming (LP)
- Feautrier’s scheduling algorithm
  - one-dimensional schedules
  - multidimensional schedules

Mathematical concepts:
- Linear Programming
- (Parametric) Integer Programming
Checking the Legality of a Schedule

Exercise: given the dependence polyhedra, check if a schedule is legal

\[ \mathcal{D}_1 : \begin{bmatrix} 1 & 1 & 0 & 0 & -1 \\ 1 & -1 & 0 & 1 & -1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 1 & -1 \\ 0 & 1 & -1 & 0 & 1 \\ 1 & -1 & 1 & 0 & -1 \end{bmatrix} \begin{bmatrix} eq \\ i_s \\ i'_s \\ n \\ 1 \end{bmatrix} \]

\[ \mathcal{D}_2 : \begin{bmatrix} 1 & 1 & 0 & 0 & -1 \\ 1 & -1 & 0 & 1 & -1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 1 & -1 \\ 0 & 1 & -1 & 0 & 1 \\ 1 & -1 & 1 & 0 & -1 \end{bmatrix} \begin{bmatrix} eq \\ i_s \\ i'_s \\ n \\ 1 \end{bmatrix} \]

1. \( \Theta = i \)
2. \( \Theta = -i \)
Checking the Legality of a Schedule

Exercise: given the dependence polyhedra, check if a schedule is legal

\[ \mathcal{D}_1 : \begin{bmatrix} 1 & 1 & 0 & 0 & -1 \\ 1 & -1 & 0 & 1 & -1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 1 & -1 \\ 0 & 1 & -1 & 0 & 1 \\ 1 & -1 & 1 & 0 & -1 \end{bmatrix} \cdot \begin{pmatrix} \text{eq} \\ i_s \\ i'_s \\ i_n \\ 1 \end{pmatrix} \quad \mathcal{D}_2 : \begin{bmatrix} 1 & 1 & 0 & 0 & -1 \\ 1 & -1 & 0 & 1 & -1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 1 & -1 \\ 0 & 1 & -1 & 0 & 1 \\ 1 & -1 & 1 & 0 & -1 \end{bmatrix} \cdot \begin{pmatrix} \text{eq} \\ i_s \\ i'_s \\ i_n \\ 1 \end{pmatrix} \]

1. \( \Theta = i \)
2. \( \Theta = -i \)

Solution: check for the emptiness of the polyhedron

\[ \mathcal{P} : \begin{bmatrix} \mathcal{D} \\ i_s \succ i'_s \end{bmatrix} \cdot \begin{pmatrix} i_s \\ i'_s \\ i_n \\ 1 \end{pmatrix} \]

where:

- \( i_s \succ i'_s \) gets the consumer instances scheduled after the producer ones
- For \( \Theta = -i \), it is \(-i_s \succ -i'_s\), which is non-empty
Reminder from Last Week

Focused on one-dimensional schedules:

- A schedule is legal if the precedence condition is respected
- It is possible to build the set of legal 1-d schedules
  - Translate the problem as finding all non-negative functions over the dependence polyhedron
  - Model them thanks to the affine form of the Farkas Lemma
  - Proceed to identification + projection to get affine constraints on the schedule coefficients

Objective for a (good) scheduling strategy

- Output a legal schedule only
- Find one which maximizes/minimizes some criterion: objective function
Linear Programming (LP)

**Definition (Linear Programming)**

Linear Programming (LP) concerns the problem of maximizing or minimizing a real-valued function over a polyhedron.

\[
\max \{cx \mid Ax \leq b\}
\]

**Theorem (Duality of Linear Programs)**

Let \( A \) be a matrix, and let \( b \) and \( c \) be vectors. Then

\[
\max \{cx \mid Ax \leq b\} = \min \{yb \mid y \geq 0, yA = c\}
\]
Equivalent Formulations

The following problems are equivalent:

1. \( \max \{ cx \mid Ax \leq b \} \)
2. \( \max \{ cx \mid x \geq 0, \ Ax \leq b \} \)
3. \( \max \{ cx \mid x \geq 0, \ Ax = b \} \)
4. \( \min \{ cx \mid Ax \geq b \} \)
5. \( \min \{ cx \mid x \geq 0, \ Ax \geq b \} \)
6. \( \min \{ cx \mid x \geq 0, \ Ax = b \} \)
7. \( \min \{ yb \mid yA \geq c \} \)
8. ...

Solving a Linear Program: Simplex Algorithm

The most standard technique: Simplex [Dantzig]

- The hyperplane $cx = v$ contains the point where the objective function has value $v$
- Optimum $v^*$ is the largest $v$ such that $cx = v$ still intersects the polytope of feasible points
- The optimum is a face of the polytope
- Simplex: starts from a vertex, and build a "path" to reach the optimal vertex
Other techniques and Complexity results

- Worst-case complexity of Simplex: exponential time $O(2^n)$
- In practice, usually around $O(n^3)$

- Ellipsoid method [Khachiyan]: worst case $O(n^4)$
- Interior points methods [Karmarkar]: worst case $O(n^{3.5})$

- LP admits a weakly polynomial-time algorithm, so LP is in $P$
Applications to Polyhedral Optimization

- LP is for real-valued objective functions
- **But we mostly use integer coefficients**
- Refinement needed: Integer Linear Programming
- Even worse: we use parametric solution sets
- We often require **Parametric Integer Programming**
Integer Linear Programming (ILP)

- ILP requires the unknown variables to be integers
- Fundamental complexity change: **ILP is NP-hard**

- Several techniques to solve an ILP: branch-and-cut, branch-and-bound, cutting planes, ...

- **Most optimization problems in the polyhedral model are modeled as ILP**
  Examples: parallelization, locality, etc.
Parametric Integer Programming (PIP)

Parametric Integer Programming [Feautrier]:

- The feasible set is parametric
- The optimal solution may not be the same for different parameter values
- PIP: "parameterized" Simplex + Gomory cuts, finds the lexicographically smallest point of a parametric polyhedron
- **Output is a Quasi-Affine Solution Tree (QUAST)**

QUAST Example: if $M = 0$ then $\{x = 0\}$ else if $M \geq 1$ then $\{x = 42\}$
Using PIPLib

- Our standard tool to solve a PIP
- Uses the same convention as PolyLib: eq/ineq on first column
- PIPLib finds the lexicosmallest point in a parametric polyhedron
  - To encode a program, add extra variables at the beginning of the system
  - These will be minimized

A few facts to keep in mind:
- The order of variables in the PIP matters (lexico-smallest is found)
- The order of parameters matters (a different solution can be found)
Back to Scheduling: Feautrier’s

Feautrier’s 1-d scheduling algorithm:

- Objective: find maximal fine-grain parallelism
- In other words: express the program loop nest as (at most) one outer sequential loop enclosing parallel loops
- This problem is equivalent to minimizing the schedule latency

Exercise: Why are the two problems equivalent?
Objective Function

Idea: bound the latency of the schedule and minimize this bound

Theorem (Schedule latency bound)

If all domains are bounded, and if there exists at least one 1-d schedule $\Theta$, then there exists at least one affine form in the structure parameters:

$$L = \vec{u} \cdot \vec{n} + w$$

such that:

$$\forall \vec{x}_R, \ L - \Theta_R(\vec{x}_R) \geq 0$$

- Objective function: $\min \{ \vec{u}, w \mid \vec{u} \cdot \vec{n} + w - \Theta \geq 0 \}$
- Subject to $\Theta$ is a legal schedule, and $\theta_i \geq 0$
- In many cases, it is equivalent to take the lexicosmallest point in the polytope of non-negative legal schedules
Example

\[
\min\{\vec{u}, w \mid \vec{u}.\vec{n} + w - \Theta \geq 0\} : \Theta_R = 0, \quad \Theta_S = k + 1
\]

Example

\begin{verbatim}
parfor (i = 0; i < N; ++i)
  parfor (j = 0; j < N; ++j)
    C[i][j] = 0;
for (k = 1; k < N + 1; ++k)
  parfor (i = 0; i < N; ++i)
    parfor (j = 0; j < N; ++j)
      C[i][j] += A[i][k-1] + B[k-1][j];
\end{verbatim}
Multidimensional Scheduling

- Some program does not have a legal 1-d schedule
- It means, it’s not possible to enforce the precedence condition for all dependences

Example

```c
for (i = 0; i < N; ++i)
  for (j = 0; j < N; ++j)
    s += s;
```

- Intuition: multidimensional time means nested time loops
- The precedence constraint needs to be adapted to multidimensional time
Dependence Satisfaction

Definition (Strong dependence satisfaction)

Given $\mathcal{D}_{R,S}$, the dependence is strongly satisfied at schedule level $k$ if

$$\forall \langle \vec{x}_R, \vec{x}_S \rangle \in \mathcal{D}_{R,S}, \quad \Theta^S_k(\vec{x}_S) - \Theta^R_k(\vec{x}_R) \geq 1$$

Definition (Weak dependence satisfaction)

Given $\mathcal{D}_{R,S}$, the dependence is weakly satisfied at dimension $k$ if

$$\forall \langle \vec{x}_R, \vec{x}_S \rangle \in \mathcal{D}_{R,S}, \quad \Theta^S_k(\vec{x}_S) - \Theta^R_k(\vec{x}_R) \geq 0$$

$$\exists \langle \vec{x}_R, \vec{x}_S \rangle \in \mathcal{D}_{R,S}, \quad \Theta^S_k(\vec{x}_S) = \Theta^R_k(\vec{x}_R)$$
Program Legality and Existence Results

- All dependence must be strongly satisfied for the program to be correct.
- **Once a dependence is strongly satisfied at level $k$, it does not contribute to the constraints of level $k + i$.**

- Unlike with 1-d schedules, it is always possible to build a legal multidimensional schedule for a SCoP [Feautrier].

---

**Theorem (Existence of an affine schedule)**

*Every static control program has a multidimensional affine schedule.*
Reformulation of the Precedence Condition

- We introduce variable \( \delta_{1}^{D_{R},S} \) to model the dependence satisfaction
- Considering the first row of the scheduling matrices, to preserve the precedence relation we have:

\[
\forall D_{R}, S, \forall \langle \bar{x}_{R}, \bar{x}_{S} \rangle \in D_{R}, S, \quad \Theta_{1}^{S}(\bar{x}_{S}) - \Theta_{1}^{R}(\bar{x}_{R}) \geq \delta_{1}^{D_{R},S}
\]

\[
\delta_{1}^{D_{R},S} \in \{0, 1\}
\]

Lemma (Semantics-preserving affine schedules)

Given a set of affine schedules \( \Theta^{R}, \Theta^{S} \ldots \) of dimension \( m \), the program semantics is preserved if:

\[
\forall D_{R}, S, \exists p \in \{1, \ldots, m\}, \quad \delta_{p}^{D_{R},S} = 1
\]

\[
\land \quad \forall j < p, \quad \delta_{j}^{D_{R},S} = 0
\]

\[
\land \quad \forall j \leq p, \forall \langle \bar{x}_{R}, \bar{x}_{S} \rangle \in D_{R}, S, \quad \Theta_{p}^{S}(\bar{x}_{S}) - \Theta_{p}^{R}(\bar{x}_{R}) \geq \delta_{j}^{D_{R},S}
\]
A Greedy Scheduling Algorithm

- Objective: maximize fine-grain parallelism
- Equivalent to strongly satisfying the maximum number of dependences at the current level
  - Find this set of schedules (objective 1)
  - Find the schedule with minimal latency in this set (objective 2)
- Proceeds greedily: removes all previously strongly solved dependence, and solve the problem for the next schedule dimension

Exercise: Write the objective function which maximizes the number of dependences strongly satisfied at a given schedule level $k$
A Greedy Scheduling Algorithm

- Objective: maximize fine-grain parallelism
- Equivalent to strongly satisfying the maximum number of dependences at the current level
  - Find this set of schedules (objective 1)
  - Find the schedule with minimal latency in this set (objective 2)
- Proceeds greedily: removes all previously strongly solved dependence, and solve the problem for the next schedule dimension

Exercise: Write the objective function which maximizes the number of dependences strongly satisfied at a given schedule level $k$

$$\max\{\sum_i \delta^i_k \mid \Theta^S_k(\vec{x}_S) - \Theta^R_k(\vec{x}_R) \geq \delta^D_{kS} \}$$
Some Interesting Properties

- Feautrier’s greedy heuristic extracts the maximal amount of fine-grain parallelism [Vivien]

- The maximal set of dependences which can be strongly solved at a given schedule level is unique

- This is true only if you do not bound the schedule coefficients

- The set of constraints to select a schedule at a given level are independent

- This formulation does not allow to build an ILP which considers multiple schedule levels, requires instead to build greedy algorithm (e.g., PluTo)
Next Week

- Building the set of all legal multidimensional schedules
- Permutability, tiling and memory optimizations
- Likely the last lecture...

In 2 weeks, I would like to have a student present a paper:

- Bondhugula, CC’08
- Irigoin and Triolet, POPL’88
- Bastoul, PACT’04
- Trifunovic, PACT’09