Handout 1:
Projection Cheat Sheet

Projection is a mechanism for increasing parallelism while maintaining logical correctness. This handout describes projection from a functional standpoint. A formal analysis of projection can be found in Projection: A Synthesis Technique for Concurrent Systems by Manohar, Lee, Martin.

The projection of a process is defined on a program template and a set of objects $S$. The program template is denoted $E(\cdot)$ such that $E(P)$ is the process in question. For many cases the template process $E(P)$ will be $\star[P]$; see the examples at the end for cases using more complex templates. An object is either a variable or a port of a channel. For channels with data the ports are represented as $C?$ and $C!$. For channels without data, read and write ports should be arbitrarily assigned. The projection results in $E(\text{proj}(S)(P))\star\text{proj}(S)(P)$, where $\text{proj}(S)$ is the projection over the set of objects $S$ and $\text{proj}(\overline{S})$ is the projection over all other objects in $P$. The projection result is a valid implementation if the projection is legal, action $P$ is executed infinitely often and the system is slack elastic.

Once a template and set of objects has been declared, evaluate the function $\text{proj}_T$ as follows:

**Elementary Statements**

Assume $A$ is an elementary statement: a skip, an assignment, or a communication.

- If $A$ has no references to objects in the set $T$ then:
  $$\text{proj}_T(A) = \text{skip}$$

- If $A$ references only objects in the set $T$ then:
  $$\text{proj}_T(A) = A$$

- If $A$ references objects in the set $T$ and objects not in the set $T$ then the projection is not legal.

**Compound Statements**

Compound statements are valid as long as the subsequent projections are legal.

$$\text{proj}_T(A; B) = \text{proj}_T(A) ; \text{proj}_T(B)$$

$$\text{proj}_T(A \parallel B) = \text{proj}_T(A) \parallel \text{proj}_T(B)$$

**Selections and Loops**

The following only describes projection for selections. Loops are handled in a similar fashion.

- If a selection $A$ references only objects in the set $T$ then:
  $$\text{proj}_T(A) = A$$

- If a selection $A$ references no objects in the set $T$ then:
  $$\text{proj}_T(A) = \text{skip}$$

- If a selection $A$ can be described as $[G_1 \land H_1 \rightarrow A_1 ; \ldots ; \land G_n \land H_n \rightarrow A_n]$ such that $G_i \equiv H_i$ (Note: $G_i$ and $H_i$ are not necessarily the same expression), and $G_i$ references only objects in the set $T_i$ and $H_i$ has no references to objects in the set $T_i$ for all $i$, then:
  $$\text{proj}_T\left([G_1 \land H_1 \rightarrow A_1 ; \ldots ; \land G_n \land H_n \rightarrow A_n]\right) = [G_1 \rightarrow \text{proj}_T(A_1) ; \ldots ; \land G_n \rightarrow \text{proj}_T(A_n)]$$

- Otherwise the projection is not legal.

Projection Examples

The following example is from homework 14:

\[
L?x; \ Y?y; \ Y!h(x); \ R!g(y) \equiv \\
\[ proj_{L?} x; \ Y!((L?x; \ Y?y; \ Y!h(x); \ R!g(y))] \\
\equiv proj_{Y?} y; \ R!(L?x; \ Y?y; \ Y!h(x); \ R!g(y)) \ \equiv \\
L?x; \ skip; \ Y!h(x); \ skip \equiv \\
L?x; \ Y!h(x) \equiv \\
\equiv \\
Y?x; \ R!g(y) \equiv \\
\equiv
\]

The following example is from *Projection: A Synthesis Technique for Concurrent Systems*.

\[
A?x; \ C?c; \\
\begin{cases}
\text{c = true} \rightarrow \ B?y; \ X!x, \ Y!y \\
\text{c = false} \rightarrow \ skip
\end{cases} \\
\equiv \\
\equiv \\
\equiv \\
\equiv
\]

\[
\text{proj}_{B?} y; \ Y!(B?y; \ X!x, \ Y!y) \equiv \\
\equiv \\
\equiv \\
\equiv
\]

Program Transformations

Often a projection over a specific set will be desired; however, the projection will not be legal since the result will contain shared variables. To prepare for projection, dependencies must be eliminated, often through copying variables. The following transformations will be useful for preparing a projection:

- Dependencies on variables and channels can often be broken by creating copies:

  \[
  E(y) \equiv \\
  Y?y', \ Y!y; \ E(y') \\
  C?y \equiv \\
  C!y; \ C'?y', \ C'!y, \ C''?y'', \ C''!y
  \]

  Notice that in the case of the channel copy, three sets should be used for projection: \((C?, \ C'!, \ C''!, \ y)\), \((C'', \ y')\) and \((C''?, \ y'')\).

- Dependencies across assignments are often broken as follows:

  \[
  x := f(S) \equiv \\
  X?x, \ X!f(S)
  \]