Motivation: Dataflow Graphs

A common Base Language

- to serve as target representation for high-level languages;
- to serve as machine language for a highly parallel machine.

Jack Dennis
Computation Structures Group, MIT
during 1967-75
**Dennis' Program Graphs**

Operators connected by arcs

- fork
- True gate (False gate)
- function or predicate

- merge

---

**Dataflow**

- Execution of an operation is *enabled* by availability of the required operand values.
- The completion of one operation makes the resulting values available to the elements of the program whose execution depends on them.

Dennis

- Execution of an operation must not cause *side-effect* to preserve *determinacy*. The effect of an operation must be local.
Firing Rules: Functional Operators

Firing Rules: T-Gate
The Switch Operator

Firing Rules: Merge
Firing Rules: Merge \textit{cont}

Some Conventions
Some Conventions Cont.

Rules To Form Dataflow Graphs: Juxtaposition
Rules To Form Dataflow Graphs: *Iteration*

Given

```
    G1
    ...  
    ...  
    ...  
    G1
    ...  
    ...  
```

Example:
The Stream Duplicator

```
    SD
    1-to-2
    SD
    
    T F
    T
    NOT
    
    T
```

November 28, 2006
http://csg.csail.mit.edu/6.827/
L20-13
The Gate Operator

Let X pass through only after C arrives.

What happens if we don’t use the gate in the Stream Duplicator?

The Stream Halver

Throws away every other token.

2-to-1 SH SH
Determinate Graphs

Graphs whose behavior is time independent, i.e., the values of output tokens are uniquely determined by the values of input tokens.

A dataflow graph formed by repeated juxtaposition and iteration of deterministic dataflow operators results in a deterministic graph.

Proof?

Dataflow Operators: Streams Functions

\[
\begin{align*}
\text{add}(x:xs,y:ys) & = +(x,y) : \text{add}(xs,ys) \\
\text{T-gate}(T:bs,x:xs) & = x: \text{T-gate}(bs,xs) \\
\text{T-gate}(F:bs,x:xs) & = \text{T-gate}(bs,xs) \\
\text{merge}(T:bs,x:xs,ys) & = x: \text{merge}(bs,xs,ys) \\
\text{merge}(F:bs,x:xs,y:ys) & = y: \text{merge}(bs,xs,ys)
\end{align*}
\]
Dataflow Graphs:
A Set of Recursive Equations

O = T-gate (A,I) ;
A = gate (I,B) ;
B = T : Not (A) ;

G. Kahn

Domain of Sequences

Sequence: [ x₁, ... , xₙ ]

The least element: [ ] (aka ⊥)

The partial order (≤): prefix order on sequences
[ ] ≤ [ x₁ ] ≤ [ x₁, x₂ ] ≤ ... ≤ [ x₁, x₂, x₃ ... xₙ ]

[ x₁, x₂, x₃ ] may be approximated by [ ] or [ x₁ ]
or[ x₁, x₂ ]. However, [ x₁, x₂ ] is a better
approximation than [ x₁ ] or [ ] for [ x₁, x₂, x₃ ].
Kleene's Iterative Solution

<table>
<thead>
<tr>
<th>I</th>
<th>[i1, i2, i3]</th>
<th>[i1, i2, i3]</th>
<th>[i1, i2, i3]</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>[ ]</td>
<td>[ T ]</td>
<td>[ T ]</td>
</tr>
<tr>
<td>B</td>
<td>[ T ]</td>
<td>[ T ]</td>
<td>[ T, F ]</td>
</tr>
<tr>
<td>O</td>
<td>[ ]</td>
<td>[ ]</td>
<td>[ i1 ]</td>
</tr>
</tbody>
</table>

O = T-gate (A, I);  
A = gate (I, B);  
B = T : Not (A);

Is the answer unique?  
Yes, if all operators are monotonic and continuous!

Monotonicity

\[ x \leq y \implies f(x) \leq f(y) \]

* a monotonic operator on sequences can only produce more output when given more input, i.e., it can never retract a value that has been produced.

* Is T-gate(B, X) monotonic?

\[ B \leq B' \implies T\text{-gate}(B, X) \leq T\text{-gate}(B', X) \]

* The proof is straightforward by the induction on the length of the sequences
Continuity

A continuous operator on sequences does not suddenly produce an output after consuming an infinite amount of input.

Is \( T\text{-gate}(B,X) \) continuous?

\[
B_0 \leq B_1 \leq \ldots \leq B_n \leq \ldots \quad \Rightarrow \\
T\text{-gate}((U_i B_i),X) = U_i T\text{-gate}(B_i,X)
\]

The proof is straightforward by the induction on the length of the sequences.

Kleene’s Fixed Point Theorem

If \( D \) is partially ordered with one least element \( (\perp) \) and is \( \omega\)-complete, and
\[
f : D \rightarrow D \text{ is a monotonic and continuous function then}
\]
\[
U_i f (\perp) \text{ is the least fixed point solution of } f.
\]

\( \omega\)-complete means that every chain \( (U_i X_i) \) in the domain has a least upper bound

\[
\perp \leq f(\perp) \leq f(f(\perp)) \leq \ldots
\]

The limit of this chain is denoted by \( U_i f_i (\perp) \)