A Data Flow Retrospective

How It All Began

Jack Dennis

MIT Computer Science
and
Artificial Intelligence Laboratory
1974 – 1975: Data Flow Years

- August 1975: 1975 Sagamore Computer Conference on Parallel Processing:
  - Rumbaugh: “Data Flow Languages”
  - Rumbaugh: “A Data Flow Multiprocessor”
  - Dennis: “Packet Communication Architecture”
  - Misunas: “Structure Processing in a Data-Flow Computer”

The symposium included a spontaneous afternoon tutorial on data flow concepts presented by Jack Dennis.
Roots

- Asynchronous Digital Logic: Muller, Bartky
- Control Structures for Parallel Programming: Conway, McIlroy, Dijkstra
- Theory of Program Schemes: Ianov, Paterson
- Structured Programming: Dijkstra, Hoare
- Functional Programming: McCarthy, Landin
Asynchronous Digital Logic - 1963

Asynchronous Logics and Application to Information Processing

DAVID E. MULLER
University of Illinois

Fig. 1

Fig. 2
Dennis: 1970
Asynchronous Control Modules

Figure 31. Control structure for channel 0 of one scoreboard section.
Programming
Control
Structures
Conway: 1963

A MULTIPROCESSOR SYSTEM DESIGN

Melvin E. Conway
Directorate of Computers, USAF
L. G. Hanscom Field
Bedford, Mass.

Figure 1. Conventions for drawing fork and join.
Coroutines: McIlroy: 1968

COROUTINES: SEMANTICS IN SEARCH OF A SYNTAX

by N. Douglas McIlroy

Oxford University and
Bell Telephone Laboratories, Incorporated.

ABSTRACT: Unlike subroutines, coroutines may be connected, and reconnected, in nonhierarchichal arrangements. Coroutines are particularly useful for generating and processing data streams. Semantics for coroutines are developed and examples are given.
Dijkstra 1965: Co-operating Sequential Processes

```
CO-OPERATING SEQUENTIAL PROCESSES

"begin integer number of queuing portions;
   number of queuing portions := 0;
parbegin
producer: begin
   again 1: produce the next portion;
      add portion to buffer;
      V(number of queuing portions);
      goto again 1
end;
consumer: begin
   again 2: P(number of queuing portions);
      take portion from buffer;
      process portion taken;
      goto again 2
end
parend
end".
```
On Ianov’s Program Schemata*

J. D. Rutledge

International Business Machines Corp., YorKtown Heights, N. Y.

Abstract. Ianov has defined a formal abstraction of the notion of program which represents the sequential and control properties of a program but suppresses the details of the operations. For these schemata he defines a notion corresponding to computation and defines equivalence of schemata in terms of it. He then gives a decision procedure for equivalence of schemata, and a deductive formalism for generating schemata equivalent to a given one. The present paper is intended, first as an exposition of Ianov’s results and simplification of his method, and second to point out certain generalizations and extensions of it. We define a somewhat generalized version of the notion of schema, in a language similar to that used in finite automata theory, and present a simple algorithm for the equivalence problem solved by Ianov. We also point out that the same problem for an extended notion of schema, considered rather briefly by Ianov, is just the equivalence problem for finite automata, which has been solved, although the decision procedure is rather long for practical use. A simple procedure for generating all schemata equivalent to a given schema is also indicated.
Paterson 1968: 
Program Schemas
Parallel Program Schemata

RICHARD M. KARP*

University of California, Berkeley, California 94720

AND

RAYMOND E. MILLER

IBM Watson Research Center, Yorktown Heights, New York 10598

Received April 15, 1968

ABSTRACT

This paper introduces a model called the parallel program schema for the representation and study of programs containing parallel sequencing. The model is related to Ianov's program schema, but extends it, both by modelling memory structure in more detail and by admitting parallel computation. The emphasis is on decision procedures, both for traditional properties, such as equivalence, and for new properties particular to parallel computation, such as determinacy and boundedness.
Karp, Miller

Parallel Program

Schema

Figure 2.
Symposium on Theoretical Programming
Novosibirsk – 1972
Notables – Novosibirsk - 1972
MIT - 1964

- IBM announces System 360.
- Project Mac selects GE 645 for Multics.
- I decide to pursue research on relation of program structure to computer architecture.
- “Machine Structures Group” formed.
Computation Structures Group: 1964 - 1975

- 1967: Jorge Rodriguez. “A Graph Model for Parallel Computations”
- 1975: Dennis, Misunas: “Preliminary Architecture for a basic Data Flow Processor”
Dennis: IFIP 1968

Fig. 3. A computation on a structure.

Fig. 4. An extended program graph.
Jorge Rodriguez
Program Graphs - 1967
Figure 27. Data-flow schema with iteration.
Dennis-Misunas Architecture 1975

Figure 14. Organization of the basic data-flow processor with auxiliary memory.
Jim Rumbaugh’s Data Flow Multiprocessor - 1975

Fig. 1. Structure of the Data Flow Multiprocessor
Related Work

- Projects at TI, ESL, Hughes, NEC, NTT, Loral
ON-LINE GRAPHICAL SPECIFICATION
OF COMPUTER PROCEDURES

W. R. SUTHERLAND

Group 23

ABSTRACT

A promising area of application for recently developed computer graphics techniques is computer programming. Two important considerations in using an interactive graphics system for drawing programs are (1) the form of a pictorial programming notation and (2) methods for making a computer execute the program once drawn. These topics are discussed in the context of an experimental graphical programming system running on the Lincoln Laboratory TX-2 Computer. This system uses a block notation for programs and can execute the drawn program with an interpreter. Improved graphical input languages for drawing programs and program notations which combine appropriate features of pictorial and written languages are needed before applications in this area are practical. The benefits to be expected from a graphical approach to programming include (1) automatic documentation, (2) debugging assistance, and (3) natural expression of parallel processes.
Enter Arvind

- 1962: Richard Kain earns MIT ScD with Project MAC and joins faculty at University of Minnesota.

- 1969: Arvind graduates from IIT Kanpur, enters U. Minn., to study Computer Science, and is inspired by Computer Architecture courses taught by Professor Kain.

- 1973: Arvind completes thesis with Professor Kain on “Models for the Comparison of Memory Management Algorithms” and joins faculty at UC Irvine.

- 1975: Arvind and Gostelow publish report on “A New Interpreter For Data Flow Schemas And Its Implications For Computer Architecture”

Workshop on
Data Flow and Reduction Languages

Department of Information and Computer Science
University of California, Irvine
Irvine, CA 92717

April 21-22, 1977

List of Invitees

J. Backus - IBM
R. Barton - Burroughs
K. Berkling - C&MD, Germany
J. D. Brock - MIT
T. C. Chen - IBM
B. Clark - Burroughs
A. Davis - Burroughs
J. Dennis - MIT
D. Friedman - Indiana University
P. Kosinski - IBM/MIT
E. Organick - University of Utah
K.-S. Weng - MIT
J. Williams - Cornell
D. Wise - Indiana University
### Workshop Schedule

#### 21 April (Thursday):

<table>
<thead>
<tr>
<th>Time</th>
<th>Presentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>AM: 9:00-10:00</td>
<td><em>Reduction Languages</em> - John Backus</td>
</tr>
<tr>
<td>10:30-11:00</td>
<td>Break</td>
</tr>
<tr>
<td>11:00-12:00</td>
<td>Discussion of Reduction Languages</td>
</tr>
<tr>
<td>12:00-1:30</td>
<td>Lunch</td>
</tr>
<tr>
<td>PM: 1:30-3:00</td>
<td><em>Data Flow</em> - Jack Dennis</td>
</tr>
<tr>
<td>1:00-3:30</td>
<td>Break</td>
</tr>
<tr>
<td>3:30-5:00</td>
<td>Discussion of Data Flow</td>
</tr>
<tr>
<td>7:00</td>
<td>Dinner</td>
</tr>
</tbody>
</table>

#### 22 April (Friday):

<table>
<thead>
<tr>
<th>Time</th>
<th>Topic for Discussion</th>
</tr>
</thead>
<tbody>
<tr>
<td>AM: 9:00-10:00</td>
<td><em>Why, or why not, data flow and reduction languages as a semantic basis for future software systems?</em></td>
</tr>
<tr>
<td>10:00-11:00</td>
<td>Discussion Break</td>
</tr>
<tr>
<td>11:00-12:00</td>
<td>Continuation of 9:00-10:00 topic, or other</td>
</tr>
<tr>
<td>12:00-1:30</td>
<td>Lunch</td>
</tr>
<tr>
<td>1:30-3:00</td>
<td>Implications for future architectures:</td>
</tr>
<tr>
<td>3:00-3:30</td>
<td>Current work</td>
</tr>
<tr>
<td>3:30-5:00</td>
<td>Break</td>
</tr>
</tbody>
</table>
1977
Data Flow and Reduction Workshop
Irvine, California
March 21-22, 1977
1977

Data Flow and Reduction Workshop
David Dennis with Gita – April 1977
Data Flow Workshop
MIT Endicott House – 1977
Computation Structures Group
Technology Square – circa 1982
Arvind: Hamilton Island, 1992