Implementing “The MATRIX”

Patrick Abela

Department of Computer Science and AI,
University of Malta

Abstract. In this technical note we shall be considering how the asynchronous constructs of the Hoopla language can be combined with concurrency constructs of parallel languages to implement a framework which allows programmers to write programs which interact with one another in a non-deterministic way. This model draws inspiration from a recently released film, The Matrix Reloaded which proposes a fiction scenario where different programs are represented by real-life characters. We shall consider how the proposed model can be used to implement a fiction game based on The Matrix or used in a wider context to implement various asynchronous activities typically occurring on a distributed environment such as the web.

1 Synchronous constructs in Sequential and Concurrent Programming Languages

Consider a simple program which is to be executed sequentially:

```
int x = 1;
int y = 2;
int z = 3;
int a = x+y;
int b = a+z;
```

Each instruction in the processor’s instruction set (assignment, conditionals etc.) is set to execute within a fixed time interval (of which duration depends on the system’s clock speed) such that it is safe for the processor to assume that as soon as a unit of time has passed it can proceed with the execution of the next instruction. Thus,the existance of a central clock mechanism and the guarantee that all instructions execute within a given unit of time allows for the sequential execution of our program. We shall be referring to this as synchronous computing in that eachinstruction is executed against a central clocking mechanism. Similarly the exection of concurrent programs is based on these same two requirements as the execution of a concurrent program involves essentially the execution of multiple sets of sequential instructions.

2 Asynchronous constructs in the Hoopla language

Hoopla is a language which was developed by the Hoopla research team at the Department of Computer Science at the University of Malta between 1997 and 1998. Hoopla programs are based on the premise that the underlying system upon which Hoopla programs are executed does not provide a central clocking mechanism and that instructions cannot be executed in a synchronous

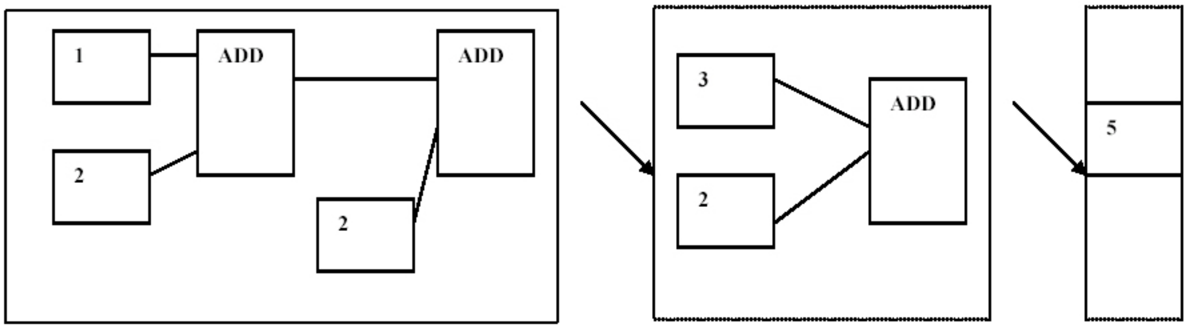

Fig. 1. Circuit reduction

manner. A virtual machine which simulates an asynchronous processor and which executes Hoopla programs was implemented as part of the research project.

Hoopla’s programming paradigm proposes programs as circuits of ‘molecular’ components which are reduced according to a set of pre-established reaction rules. Any group of interconnected molecules can be reduced if an applicable reaction rule is found. Figure 1 illustrates how a program with three interconnected molecular components is evaluated. Note that in this simple example the second addition cannot occur before the evaluation of the first addition. This is called ’causal sequentiality’and demonstrates one way of how sequential constructs can be implemented in such a paradigm.

The Hoopla language proposed a set of twelve molecules which could be used to express assignments, conditionals and all other constructs used for flow control. Most of the business logic used for the reduction of these circuits was implemented in the virtual machine itself such that new molecules could not be added easily. A revised version of the virtual machine was subsequently implemented such that the business logic for the interaction between the molecules was contained within the molecule components. These ’micro-program molecules’ allowed the virtual machine to scale to support new molecule types. Various molecule sets were proposed and implemented on this virtual machine.

3 The Matrix: Reloaded

The recently released film, Matrix Reloaded presents an animated depiction of intelligent programs which act autonomously and which interact with one another. These programs interact and challenge one another through various techniques most of which and well known to programmers.For instance in one instance, a program clones itself repeatedly and attacks another program in what can be considered a typical ’denial of service’ attack.

It is our idea to build a game whereby programmers could write their programs as molecules which are then launched to react asynchronously with other molecules released by other programmers. The game would be distributed on the web and the virtual machine would have the capability of reducing molecule circuits which span across a distributed environment. Reaction rules would be implemented by the programmers within the molecules.

4 Extending the Hoopla asynchronous model

A new model is being proposed to address the requirements of such an application. This model makes use of Hoopla's asynchronous programming paradigm to have various components react with one another. Once two or more molecules are ready to react communication channels similar to those used in parallel languages are established such that information can be exchanged.

In order to make the game more realistic we might extend the model such that molecules or micro-programs are associated with a position in a 3D co-ordinate system. Micro programs will react with one another if they share the same location. Using this notion it would be possible to build 3D defence systems which make use of 'Wall' molecules and 'Door' molecules.Similarly it would be also possible to implement special types of molecules which 'spy' on messages which are being sent as molecules between different sites. To counter balance this the programmers might need to implement special molecules which make use of keys to encrypt information.