한국시뮬레이션학회 '99춘계학술대회 논문집 1999.4.17 동의대학교

GUI를 이용한 병렬/분산 시뮬레이션 환경의 개발

Development of a GUI-based Parallel and Distributed Simulation Environment

†강원석, †사공봉,*홍준성, †김기형
*삼성전자 중앙연구소
†영남대학교 컴퓨터공학과

요 약

병렬/분산 시뮬레이션은 시뮬레이션의 수행 속도를 높이려는 시도로서 많이 연구되고 있는 분야이다. 기존의 병렬/분산시뮬레이션 연구에서는 주로 시뮬레이션 속도를 향상시키려는 동기화 기법에 대해 연구의 초점이 맞추어져 왔다. 따라서 전문가가 아니면 병렬 모델을 만들기 매우 힘든 문제점이 있었다. 본 논문에서는 병렬/분산 시뮬레이션을 위한 GUI 기반의 모델링 및 시뮬레이션 환경을 제시한다. 제시된 환경에서는 사용자가 모델기술을 GUI기반으로 쉽게 할 수 있고 병렬 시뮬레이션을 위해 모델을 자동으로 파티션 및 수행 코드를 발생시켜준다. 따라서 비전문가도 쉽게 분산 시뮬레이션을 할 수 있도록 자동화 할 수 있다.

1. 서론

1)시뮬레이션은 시스템설계 및 검증을 포함한 많은 응용 분야에서 사용되어 왔다. 대규모의 시뮬레이션을 하려면 방대한 시뮬레이션 모델을 구성해야 하고 또한 시뮬레이션의 수행에 많은 시간이 걸리게 된다. 이 중에서도 시뮬레이션 수행속도를 줄이려는 시도로서 병렬/분산 시뮬레이션에 대한 연구가 많이 진행되어 왔다[4,5,6]. 하지만 병렬/분산 환경에서의 시뮬레이션 모델 저작은 전문가가 아니면 매우 힘들기 때문에 병렬/분산 시뮬레이션의 보급에 가장 큰 걸림돌이 되어 왔다.

본 논문에서는 GUI기반의 모델링 및 시뮬레이션 환경을 제시한다. 제시된 환경에서는 비주얼 모델링이 가능하여 보다 쉽게 모델링을 할 수 있으며, 병렬/분산 시뮬레이션 코드를 발생시켜준다. 이를 위해 DEVS 형식론에 기반 하였으며 사용자는 계층적이고 모듈화된 모델을 작성할 수 있다. 제시된 환경이 비주얼 모델링의 첫 번째 시도는 아

1) 본 논문은 1998년도 정보통신부 대학기초연구지원사업에 의하여 연구되었음.

니다. event graph, activity cycle, control flow graph등이 그 시도들이다. 그러나 이들은 분산 시뮬레이션을 위해 만들어지지는 않았다.

제시된 환경은 병렬/분산 시뮬레이션을 위해 자동으로 모델을 파티션하는 알고리즘[7]을 구현하였으며 사용자는 쉽게 파티션 결과를 볼 수 있다.

2. DEVS 형식론

DEVS는 이산사건 시스템을 위한 집합이론에 기반한 형식론이다. DEVS 모델은 실시스템의 동작을 기술하는 원소(atomic) 모델과 연결(coupled) 모델로 구분되며, 이 두가지 모델을 사용하여 계층적이고 모듈화된 형태로 모델구성이 가능하다.

$AM=\langle X, Y, S, \delta_{ext}, \delta_{int}, \lambda, ta\rangle$

단,

X: 외부 입력사건의 집합

Y: 출력 사건의 집합

S: 순차 상태의 집합

$\delta_{ext}: Q\times X\rightarrow S$: 외부 상태 천이 함수

$Q=\{(s, e)|s\in S \text{and} 0\le e\le ta(s)\}$

$\delta_{int}: S\rightarrow S$: 내부 상태천이함수

$\lambda: S\rightarrow Y$: 출력함수

$ta: S\rightarrow R^{+}_{0,\infty}$: 시간 진행함수

원소 모델 AM은 외부 입력 사건 x에 의해 상태 천이($s'=\delta_{ext}(s)$)를 일으키며 이에 대한 반응으로 시간진행함수 $ta(s')$시간 후에 출력 사건 y를 일으키게 된다. 이때 모델의 상태는 집합 S의 원소 s로 표현된다. 출력 사건 Y를 일으킨 후에는 내부 상태 천이를 하게 되는데 이 함수가 δ_{int} 함수이다. 그러나 만일 ta(s')이전에 다른 외부 입력 사건 x'을 받게 되면 새로운 외부 상태천이를 하게 되며 새로운 사이클이 진행되게 된다.

연결 모델은 모델의 결합 및 연결 상태를 기술하게 된다.

$CM=\langle D, \{M_i\}, \{I_i\}, \{Z_{i,j}\}, SELECT\rangle$

단,

D: 컴포넌트이름 집합

For each i in D

M_i : 컴포넌트 모델

I_i: i의 출력에 연결된 컴포넌트 집합

For each j in I_i

$Z_{i,j}: Y_i\rightarrow X_j$: i to j 출력 변환

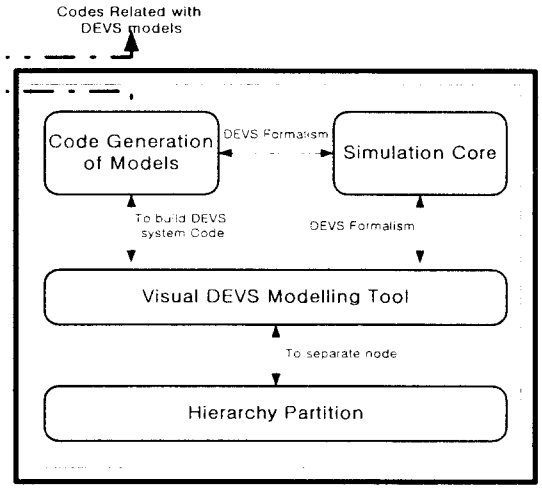
$SELECT: 2^M-\phi\rightarrow M$: 같은 시간에 내부 상태 천이를 하려는 모델중 우선순위를 정해주는 함수

연결 모델 CM은 컴포넌트(원소모델이나 또다른 연결모델이 될 수 있다) 구성되며, $\{I_i\}$와 $\{Z_{i,j}\}$는 컴포넌트들간의 연결을 나타낸다.

연결 모델과 원소모델을 이용하여 사용자는 계층적인 모델 구성을 할 수 있고 수학적 형식론에 기반함으로써 모델 재사용, 논리적 분석등의 기법을 활용할 수 있는 장점을 가진다.

3. 병렬/분산 모델링 개발 툴

(그림 1)과 같이 병렬/분산 모델링 개발



(그림 1) 병렬/분산 모델링 개발툴 구성도

툴은 Code Generation, Simulation Core, Visual DEVS Modelling Tool, Hierarchy Partition으로 구성되어 있다.

Code Generation은 병렬/분산 모델링을 구성하였을 때 PDEVS 시뮬레이션 엔진에 실행할 수 있는 코드를 생성한다. Simulation Core는 DEVS에 기반한 Atomic과 Coupled모델들의 정보를 구성하고 있다. Visual DEVS Modelling Tool은 모델을 GUI기반으로 구성하는 툴이다. Hierarchy Partition은 모델들을 분산 시 모델들에 대한 프로세서 ID를 제공한다.

3.1 Code Generation

Code Generation은 PDEVS 시뮬레이션 엔진에 포팅을 위한 C 코드를 생성한다. PDEVS 시뮬레이션 엔진에 포팅을 위해 필요한 함수는 5가지가 있다[1]. 이들 함수는 아래의 <표 1>와 같다.

<표 1> Code Generator에서 생성되는 코드

종류	함수 Type
Constructor	create_(Model_Name)(...)
External func.	(Model_Name)_ext_transfn(...)
Internal func.	(Model_Name)_int_transfn(...)
Output func.	(Model_Name)_outputfn(...)
Time advance func.	(Model_Name)_time_advancefn(...)

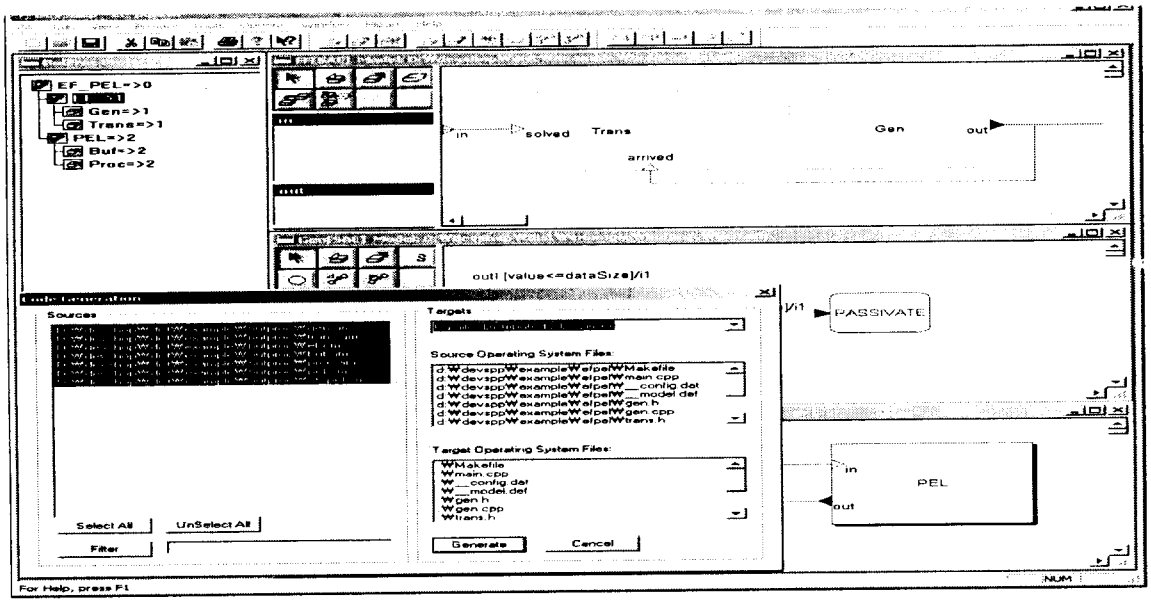
Constructor는 시뮬레이션 엔진이 구동시 모델을 생성할 때 처리되는 함수로 State, ta를 초기화한다. External function과 Internal function은 모델에 Message Event가 도착했을 실행되는 함수이고, Output function은 모델 포트에 어떤 Message를 보낸다. 마지막으로 Time advance function은 모델이 수행되고 있는 시간을 결정한다. 또한 PDEVS 시뮬레이션 시스템에서 모델들의 정보를 생성한다. 이들 정보는 모델링 입출력 포트 정보와 분할된 모델의 프로세서 ID와 모델들간의 입출력 포트로 구성된다.

3.2 Simulation Core

Simulation Core는 DEVS형식론에 기반한 이산사건 모델을 계층화되어 구성되어진다. DEVS형식론은 Atomic 모델과 Coupled 모델로 구성되어 있다. Atomic 모델은 시스템의 행동을 기술하며, Coupled 모델은 자신의 서브모델들의 연결정보를 나타낸다[2].

3.3 Visual DEVS Modelling Tool(VDMT)

VDMT는 GUI환경의 모델링 개발 툴이다. VDMT는 모델들 각각에 대한 external, internal transition, Output, Phase, State 변수, 입출력 포트 정보를 각각의 모델들에 적용한다. (그림 2)는 VDMT의 실행화면이다.

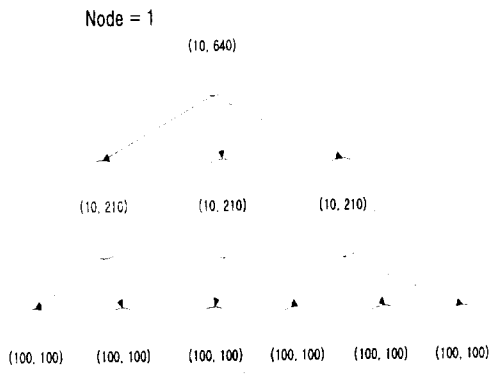


(그림 2) Visual DEVS Modelling Tool

3.4 Hierarchy Partition

Hierarchy Partition은 계층화된 DEVS 모델을 PDEVS에 적용시키기 위한 프로세서 ID를 생성해준다. 모델 분할 방법은 아래에서 설명한다. 각각의 모델은 자신의 Cost와 하부 모델들의 전체 Cost를 가진다**[3]**. 각각의 모델에 (Cur_Cost, Sum_Below_Cost)와 같은 집합 정보가 저장된다. Cur_Cost는 자신의 Cost이며 Sum_Below_Cost는 하부 모델들의 전체 Cost이다. 각각 Coupled 모델은 Cost를 10으로, Atomic 모델은 100으로 하자.

(그림 3)과 같이 모델들이 구성되어 있다고 하면 Root 모델의 정보는 (10, 640)이 된다.



(그림 3) 로컬머신에서 실행(Node 1개)

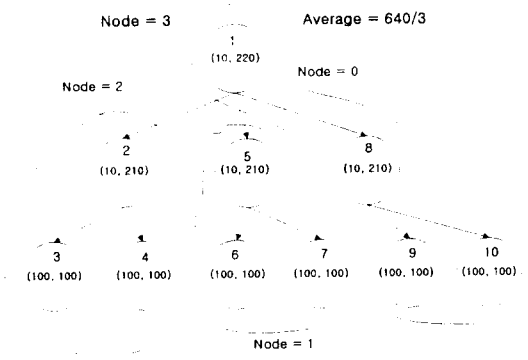
(그림 3)은 로컬 머신에서 실행될 때를 나타낸다. PDEVS 시뮬레이션 시스템에 이점을 살려 이들 모델들을 분할하면 (그림 4)와 같다. (그림 4)는 3개의 모델로 분할된 경우이다. 분할 과정은 다음과 같다.

① 모델들의 총 Cost를 분할할 개수로 나누어 Average Cost를 구한다.

② Root 모델의 자식노드들을 찾으면서 Average Cost와 비교하여 일정 범위 [*Average - (Average/m) <= Sum_Below_Cost <= Average + (Average/m)*]이내에 들어가면 모델을 분할한다.

③ 2번 모델에 방문했을 때 Average Cost ≒ 210 이고 Sum_Below_Cost는 210이기에 하나의 노드로 분할된다.

④ 위의 과정을 반복하여 노드를 분할한다.



(그림 4) 모델을 3개의 프로세스로 분할

VMDT는 각각의 모델들에 대한 임임의 Cost를 입력할 수 있다. 또한 모델 분할 시 프로세서 ID를 직접 바꿀 수 있다.

4. 결론 및 향후계획

본 논문에서는 병렬분산시뮬레이션을 위한 GUI기반의 모델링 환경을 제시하였다. 제시된 환경은 비주얼 모델링이 가능하며 병렬/분산 시뮬레이션을 위해 모델 파티션 및 코드 발생을 자동으로 해준다. 또한 순차 시뮬레이션코드로도 발생시켜주므로 사용자는 순차시뮬레이션으로 모델의 검증을 하고 속도향상을 위하여 병렬/분산 시뮬레이션을 활용할 수 있다.

앞으로 시뮬레이션 엔진을 본 시스템과 직접 연결함으로써 개발 툴뿐만 아니라 테스팅 환경으로도 구축 가능 할 것이다.

참고문헌

[1] 성영락, 정성훈, 김탁곤, 박규호, "병렬분산 환경에서의 DEVS형식론의 구현", 한국시뮬레이션학회 논문지, Vol.1, no.1, pp.64-76, 1992.

[2] Zeigler, B.P 1984. Multifacetted Modelling and Discrete Event Simulation. London: Academic Press.

[3] 김기형, "시스템 이론적 형식론에 기반한 분산 시뮬레이션 방법론 : 비동기적 접근", 박사학위논문, 1996.

[4] Fujimoto, R.M., "Parallel discrete event simulation," Commun. ACM, vol.33, no.10, pp33-53, 1990

[5] Jefferson, D., "Virtual time," ACM Transactions on Programming Languages and Systems, vol.7, no. 3 pp404-425, 1985

[6] Jha, V. and Bagrodia, R.L., "A unified framework for conservative and optimistic distributed simulation," In proceedings of the 8th workshop on parallel and distributed simulation," Simulation Councils, Inc

[7] Kihyung Kim, Tag gon Kim, Kyu ho Park, "Hierarchical partitioning algorithm for optimistic distributed simulation of DEVS models," Journal of Systems Architecture, pp433-455, 1998