그래프 중간표현 형태를 기반으로 한 병렬 프로그래밍 환경의 설계 및 구현

이원용[0] 박두순* 송상주**
[0]혜전대학 컴퓨터 계열 조교수, *순천향대학교 정보기술공학부 교수 **순천향대학교 전산학과 박사과정

A Design and Implementation of Parallel Programming Environment using Graph Type Intermediate Representation Form

Won-Yong Lee[0] Doo-Soon Park* Sang-Ju Song**
Hyejeon College Computer Division, Soonchunhyang University Division of Information & Technology Engineering

요 약

본 논문에서는 사용자의 병렬 프로그램 작성을 도와주는 병렬 프로그래밍 환경을 제공한다. 병렬 프로그램은 다양한 하드웨어의 특성에 따라 또는 프로그램의 특성에 따라 사용자가 병렬 프로그램을 작성하여야 하기 때문에 병렬프로그램을 작성하는 것이 매우 어렵다. 본 논문에서는 많은 병렬화 연구에서 제시되고 있는 그래프 중간 표현 형태를 그래프 사용자 인터페이스로 구현하였다. 이 병렬 환경에서는 프로그램 편집기능, 종속성 분석기능, 루프 변환기능, CFG, PDG, HTG 등 중간 코드를 그래프 중간 표현 형태를 통해 보여 줌으로 최적의 병렬프로그래밍 환경을 제공한다.

1. 서 론

정보산업의 발전에 따라 컴퓨터의 응용분야는 넓어지고 컴퓨터가 처리해야 할 정보의 양은 점점 방대하여 지고 있다. 이에 따라 좀더 빠른 시간에 많은 양의 정보를 처리 할 수 있도록 연구가 진행되고 있다. 이 방대한 양의 정보를 처리하기 위하여 병렬 컴퓨터가 등장하게 되었으나 병렬 컴퓨터를 사용하는 과정에서 많은 어려움이 발생되고 있다. 본 논문에서는 이 병렬컴퓨터 환경에서 사용자가 쉽게 프로그램을 할 수 있도록 도와주는 환경을 제공한다.[1]

본 논문은 그래프 중간 표현 형태를 기반으로 한 병렬프로그램 환경은 기존 순차 프로그램을 입력받아 입력 된 순차 프로그램의 중간코드를 그래픽 인터페이스를 이용하여 그래프 형태로 사용자 정보를 제공하므로 좀더 효과적이고, 효율적인 프로그램 작성을 도와주는 것을 목적으로 한다. 이를 위하여 그래픽 사용자 인터페이스를 설계, 제안하고 이에 따라 일련의 작업을 수행하면서 중간코드에서 발생되는 여러 정보를 사용자에게 그래프를 통하여 변환과정을 쉽게 분석, 이용할 수 있도록 환경을 제공한다.

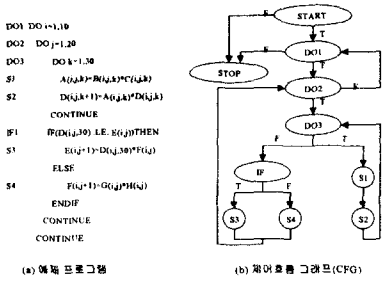
2. 병렬 프로그래밍의 환경

일반적으로 병렬프로그래밍의 환경은 어휘분석, 구문분석, 자료흐름분석, 자료 종속성 분석, 병렬화 프로그램 변환 과정을 거쳐 병렬프로그램을 얻을 수 있다. 본 장에서는 여러 가지 그래프 중간 표현 형태들과 병렬화에 대해 기술한다.

2.1 CFG(Control Flow Graph: 제어흐름그래프)

CFG는 컴파일러 과정에서 제어 종속성을 분석하는 과정에서 제어 종속성을 표현하기 위하여 사용하며 흐름 그래프로부터 유도된다. CFG는 제어흐름에 의한 프로그램의 방향성 그래프(directed graph)로서 G=(V, E) 형태로 나타내는데, 여기서 V는 노드(node)를 나타내고 이 노드는 프로그램의 기본 블록이나 한 문장단위를 나타낸다. E는 연결선을 나타내는데 두 노드 사이의 제어흐름을 나타낸다. 제어흐름 그래프는 한 개의 START 노드와 STOP 노드를 갖는다.

(그림 2.1)은 예제 프로그램과 CFG를 나타낸 것이다.



(a) 예제 프로그램 (b) 제어흐름 그래프(CFG)

(그림 2.1) 예제 프로그램과 CFG

2.2. PDG(Program Dependence Graph: 프로그램 종속성 그래프)

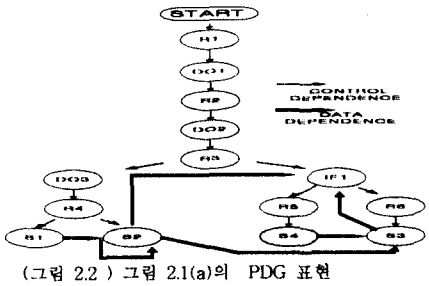
PDG는 CFG로부터 유도할 수 있는데 PDG를 유도하기 위해서는 먼저 제어종속 부그래프(control dependence subgraph)와 자료 종속 부그래프(data dependence subgraph)를 유도하여야 한다[2,3]. 이 두 그래프를 합하여 제어종속과 자료종속을 동시에 표현한다.(그림 2.1)의 예제 프로그램에서 PDG를 유도하기 위해서는 먼저 CFG를 유도하고 이 유도된 CFG에서 노드들간의 자료종속성 관계를 추출한 다음 기본 블록에 제어종속성을 나타내는 연결선으로 연결하여 제어종속 부 그래프를 구성한다. (그림 2.2)는 (그림 2.1(a)의 원시프로그램에서 유도된 PDG를 보여준다.

2.3. HTG (Hierarchical Task Graph: 계층적 태스크 그래프)

Girkir에 의해 제안된 HTG는 순차 프로그램으로부터 함수형 병렬성을 유도하는 중간 그래프이다. 이 그래프는 계층적 태스크 루프의 개념을 도입하고 자료 제어간의 종속성을 제거하여 중간 그래프를 완성하였다.

HTG는 구조적 프로그램에서 유도할 수 있으며 프로그램을 줄일 수 있다는 것은 루프에 기반을 둔 수직적 구조로 쉽게 표현될 수 있고, 단일 입구와 출구가 있는 반복수행 단위를 갖는 루프이다[4].

HTG는 CFG로부터 유도되는데 CFG를 DFS(Depth First Search)알고



(그림 2.2) 그림 2.1(a)의 PDG 표현

리즘의 수행하여 그와 관계된 경로를 계층화 하여 HTG를 구축한다[4].

2.4 루프 변환

원시 입력 프로그램 내에서 다중 루프가 존재할 경우 루프 안쪽에 위치하고 있는 루프가 벡터화가 될 수 없고, 비효율적인 벡터코드가 생성될 경우가 발생된다. 이런 경우 원시 프로그램의 의미가 손상되지 않는 범위 안에서 루프의 수행 순서를 변경할 수 있다[5,6].

루프교환을 구현하기 위해서는 각각의 루프 인덱스 역할을 하는 숫자를 부여하고 그리고 종속성 정보를 나타내는 방향벡터 중에서 각각의 루프의 인덱스에 해당되는 부분을 찾아내 루프교환여부가 가능한지를 판단하여 루프 교환을 한다. 루프융합은 이웃하고 있는 두 루프를 하나의 루프로 변환하는 것으로 루프 변수의 영역이 동일해야하며 융합했을 때 자료 종속성 정보가 변하지 말아야 한다. 루프분산은 하나의 루프안에 있는 여러개의 문장들을 자료 종속관계에 따라 좀더 작은 단위로 분할하는 방법이다[17].

3. 병렬 프로그래밍 환경 구현

본 장에서는 그래프 중간 표현 형태를 기반으로 한 병렬프로그래밍 환경에 대하여 소개하고 그 기능 및 작동방법에 대하여 기술한다. (그림 3.1)은 그래프 중간 표현 형태를 기반으로 한 병렬 프로그래밍 환경을 보여주고 있다.

3.1 사용자 환경

본 병렬프로그래밍 환경은 사용자와 병렬 프로그램 환경사이에 상호 작용이 용이하기 위해 X Window를 기반으로 그래픽 인터페이스를 사용하며, 메뉴 선택방식을 이용하여 병렬 프로그래밍 환경을 제공하였다. 본 논문에서 제공하는 병렬 환경은 특정 병렬 컴퓨터 및 하드웨어, 소프트웨어와는 무관하게 일반 컴퓨터에서 병렬 프로그램을 용이하게 사용하도록 설계하였다. (표 3.1)는 사용자 메뉴와 기능을 나타낸다.

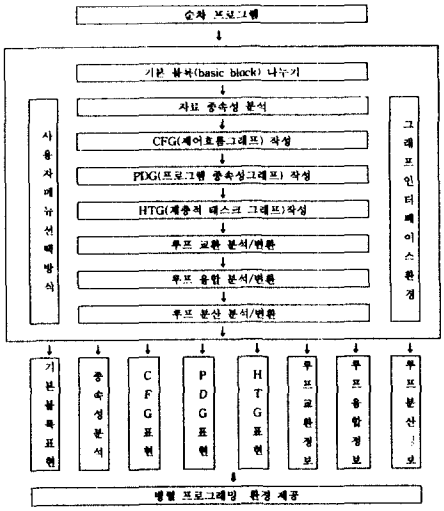
3.2 병렬 프로그래밍 환경 구현

병렬 프로그래밍 환경에서 사용언어는 C 언어로 구현하였으며 win 98 운영체제를 사용하여 펜티엄 II에서 구현하였다.

단말에서 "icgen"을 수행하면 병렬 환경 프로그래밍 메뉴가 떠오른다. 사용자는 병렬프로그램을 작성하기 위하여 newfile을 선택하면 파일을 편집할 수 있는 창이 나타난다. 사용자는 이 병렬 프로그램을 작성한 후 save라는 명령단추를 선택하면 icdir로 프로그램이 저장된다. 이 프로그램 저장시 특정한 프로그램이름을 주지 않을 경우는 icgen.c로 저장된다.

이 과정에서 사용자는 파일 편집기능을 이용하여 작업중의 내용을 편집, 수정할 수 있다.(그림 3.2)는 병렬환경의 초기 메뉴창을 보여준다.

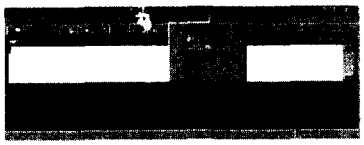
종속성분석은 특정 프로그램을 선택하여 종속성 관계를 분석하고자 할 때 사용한다. 사용자는 종속성 분석을 하기 위해서는 먼저 분석 하고자 하는 프로그램을 오픈하고 해당 프로그램을 로드 시키고 이 로드된 프로그램을 종속성 그래프 명령단추를 입력하면 분석기는 프로그램 내에서 어휘분석과



(그림 3.1) 그래프 중간 표현 형태를 기반으로 한 병렬 프로그래밍 환경

(표 3. 1) 사용자 메뉴

작업구분	메뉴	작업 기능
파일	newfile	새프로그램 입력
	open	기존 프로그램 열기
	load	기존 프로그램 수정
	save	프로그램 저장
	save as	다른이름으로 저장
	quit	프로그램 종료
편집	cut	편집내용 일부삭제
	add	편집내용 덧붙임
	delete	프로그램 삭제
기본블록	기본블록	기본 블록 나누기
종속성	종속성 분석	종속성 정보 출력
		종속성 그래프 표현
중간 그래프	CFG	제어 흐름 그래프
	PDG	프로그램 종속성그래프
	HTG	계층적 태스크 그래프
루프 변환	loop interchange	루프 교환
	loop distribution	루프 분산
	loop fusion	루프 융합
출력	화면내용 출력	
도움말	작업내용 도움말	

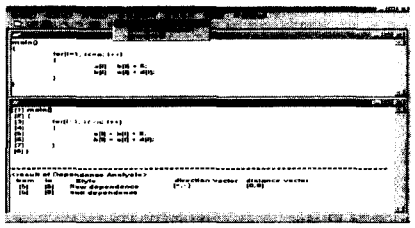


(그림 3.2) 초기 메뉴창의 모습

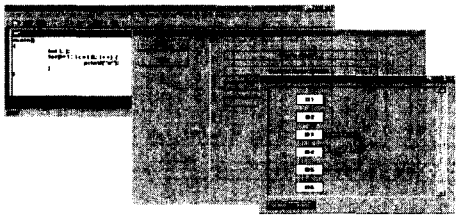
구분분석을 수행한 후 자료 흐름분석과 종속성 분석을 수행한다. 이 종속성 분석은 현재 계산된 값을 보관하고 프로그램이 종료된 후 변경값을 비교하여 문장간의 종속성 정보를 나타낸다.

(그림 3.3)는 특정 프로그램을 종속성분석을 하였을 경우의 창의 모습을 보여주고 있다.

CFG는 기본 블록에서 제어 정보를 추가하여 그래프로 표현한 것이다. 이 그래프는 병렬 프로그램을 작성하여 CFG 명령 단추를 입력하면 화면에 CFG가 작성되어 창에 보여준다. (그림 3.4)는 순차 프로그램을 보여주고 이 프로그램에 의해 작성된 CFG를 보여주고 있다.

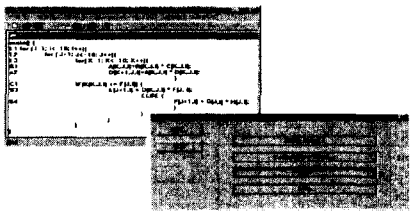


(그림 3.3) 종속성 분석 결과

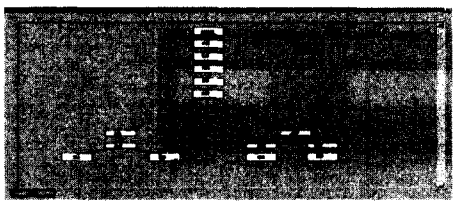


(그림 3.4) CFG 의 표현

PDG는 제어 종속성 정보와 자료 종속성 정보를 한 그래프에 표시한 중간 그래프이다. 이 그래프를 표현하기 위해서는 문장을 기본 블록으로 나누어 CFG를 작성한 후 노드간의 자료종속성, 자료 종속성을 분석하여 그 정보를 각각 한 그래프를 표현한다. (그림 3.5)과 (그림 3.6)은 예제 프로그램과 PDG를 나타낸 창이다.



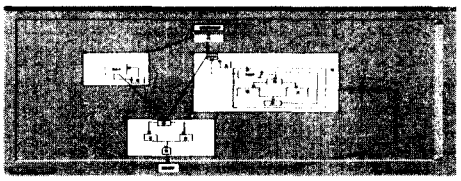
(그림 3.5)의 PDG의 예제 프로그램과 메뉴창



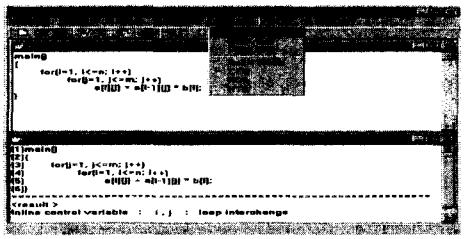
(그림 3.6) PDG 표현

HTG는 루프 노드를 계층적으로 표현한 중간 그래프이다. 이 HTG는 구조적 프로그램형태로 구성되어야 하는데 이 HTG를 표현하기 위해서는 먼저 CFG를 작성한 후 이를 DFS알고리즘의 수행하여 후의 경로 추출하여 작성한다. (그림 3.7)은 HTG를 구현한 그래프를 나타내고 있다.

루프교환은 사용자가 먼저 해당 프로그램을 load하여 종속성 분석 메뉴를 이용하여 종속성 여부를 파악한 후에 루프 교환 가능성을 판단한다. 루프 교환은 원시 프로그램의 의미가 손상되지 않도록 루프의 수행 순서를 바꾸는 것이다. 사용자는 루프 교환 여부를 파악한 후 대단위 병렬화가 목적이라면 관련 첨자를 바꿀 수 있다. (그림 3.8)는 변경전, 후의 창의 모습을 보여주고 있다.



(그림 3.7) HTG의 표현



(그림 3.8) 루프 교환 전, 후의 창의 모습

5. 결론

본 논문에서는 일반적인 병렬 프로그래밍 환경에 대해 소개하고, 병렬 프로그래밍 환경을 제공하기 위한 그래프 중간 표현 형태를 기반으로 한 병렬 프로그래밍 환경을 설계하고 구현하였다. 본 논문에서 제공하는 병렬환경은 중간 그래프를 기반으로 한 병렬 프로그래밍 환경에서 일반 편집기에서 제공하는 모든 기능을 X Window 환경에서 그래픽 인터페이스 방식과 메뉴선택방식을 이용하여 사용자가 쉽게 사용할 수 있도록 설계 구현하였다. 그리고 중간 표현 형태를 기반으로 한 병렬 프로그래밍 환경에서는 순차 프로그램을 입력하여 이 프로그램을 기본 블록으로 나누고, 자료 종속성을 분석하고 정보를 제공한다. 또 원시 프로그램을 입력하여 CFG, PDG, HTG를 사용자에게 중간 표현 형태로 그래프를 제공하며, 루프 변환 정보를 사용자에게 제공한다.

참고문헌

[1] Allen J. R. and K. Kennedy, "A Parallel Programming Environment," IEEE Software, Vol 2, No 4, pp 21-29, July, 1985

[2] Ferrante J., K. J. Ottenstein and J. Warren, "The Program Dependence Graph and Its Use in Optimization," ACM Tran. on Programming Language and System, Vol. 9, NO. 3, pp 319-349, July 1987.

[3] 박성순 , 그래프 중간 표현 형태를 이용한 프로그램 벡타화, 고려대학교, 박사학위논문 논문, 1993

[4] Girkar, Milind Baburao, "Functional Parallelism: Theoretical Foundations and Implemention" Ph. D. These, University of Illions at Urbana-Champaign. , 1992

[5] 송월봉, 박두순, "최대 병렬성 추출을 위한 자료 종속성 제거 알고리즘", 정보과학회 논문지 제 26권 제 1호, pp 139-149, 1999

[6] 이만호,"병렬화를 위한 루프의 구조의 변환", 정보과학회지 제 12권 제 5호, pp 54-56, June, 1994