트랜스퓨터에서의 효율적인 병렬처리에 관한 연구

김 영희[0] , 박 두순
순천향 대학교 공과대학 컴퓨터학부

A Study on effective parallel processing in Transputer

Young-Hee Kim[0], Doo-Soon Park
Department of Computer Science, College of Engineering
Soonchungyang University

요 약

병렬처리 컴퓨터는 하드웨어, 소프트웨어적인 두 가지 측면에서 동시에 만족되어질 때 최적의 성능 향상을 가져올 수 있다. 본 연구는 다양한 토폴로지를 제공하고 가격대 성능비가 좋은 트랜스퓨터상에서 자료간 종속 관계에 있는 병렬 코드를 수행하는 방법들을 소프트웨어적인 기법을 통해 알아보고 종속 관계에 있는 자료 처리 시 프로세서 수의 증가를 통한 속도향상을 실험하였다. 그 결과 본 논문에서 제시한 코드로 자료의 교환량을 최소화하기 위한 기법인 경우 프로세서의 수가 2개 일 때 cost-effective임을 제시하였다. 따라서 트랜스퓨터에서 효율적인 병렬 처리를 위해서는 각 node의 토폴로지, 자료분산 모델, processor의 개수들이 반드시 고려되어야 한다.

1. 서 론

병렬처리(Parallel processing)컴퓨터는 순차처리보다 처리속도를 빠르게 하고 많은 양의 자료를 단일프로세서가 처리하지 못할 경우 독립된 자료들을 여러개의 프로세서에게 나누어 처리하므로 대용량의 처리를 가능하게 할 뿐 만 아니라 시스템의 성능향상을 가져 올 수 있다. 이러한 병렬처리를 하기 위해서는 순차처리에서 고려되지 않아도 되는 많은 문제가 야기되고 이러한 문제는 하드웨어적인 측면과 소프트웨어적인 측면으로 나누어 생각할 수 있다.

하드웨어적인 측면에서 고가의 빠른 단일 프로세서 대신 값이 저렴한 다중 프로세서 여러개를 사용하여 다양한 프로세서 망의 구성을 통해 강력한 병렬 처리 시스템의 구현을 가능하게 한다. 또한, 병렬처리 컴퓨터의 특성을 가능한 최대로 발휘하도록 하며 시스템 이용률을 극대화 할 수 있는 여러 가지 소프트웨어적인 측면이 고려되어지고 있다. 이들은 어떤 문제 해결에 있어서 두 가지 측면이 동시에 만족되어질 때 가장 최적의 성능 향상을 가져올 수 있다. 본 논문은 이러한 두 가지 측면을 고려하여 효율적인 병렬처리를 하고 문제에 대한 병렬성을 극대화 시키고자 여러개의 프로세서를 추가하여 병렬처리 컴퓨터의 구성이 가능하고 프로세서간 다양한 토폴로지를 제공하므로 여러 가지 응용 프로그램에 맞게 시스템 변경이 가능한 트랜스퓨터를 이용하여 다수의 프로세서에게 작업 분배시 시스템의 이용률을 증대 시킬 수 있는 자료분산 모델들의 특징과 문

제 분할, 스케쥴링, 동기화, 병렬알고리즘과 같은 소프트웨어 기법을 알아보았다. 다음으로 위의 내용들을 이용하여 병렬처리 하고자 재구성컴파일러 알고리즘에서 추출된 병렬코드를 이용하여 트랜스퓨터에서 수행시키고, 수행 결과를 분석하여 효율적인 소프트웨어 기법들을 제시한다.

본 논문은 제 2장에서는 본 논문에서 병렬처리 컴퓨터로 사용된 트랜스퓨터와 트랜스퓨터에서의 자료분산 모델들의 특징을 알아 보았다. 제 3장에서는 병렬처리 기법 중 소프트웨어적인 기법과 트랜스퓨터를 이용한 토폴로지의 구성 방법을 알아 봄으로써 병렬처리가 가능하도록 하였고 제 4장에서는 트랜스퓨터에서 수행할 병렬코드를 이용하여 시스템 토폴로지 구성, 자료분산 모델을 통한 자료 분배의 실험모델을 설정하였다. 제 5장에서는 4장에서 설정된 모델을 이용해 트랜스퓨터에서 실행한 후 실제 수행 시간을 알아보고 성능을 평가해 봄으로써 더 효율적인 병렬처리 방법을 제시했다. 제 6장은 결론과 추구연구 과제에 대하여 기술하였다.

2. 트랜스퓨터의 특징 및 자료분산 모델

트랜스퓨터는 메시지 교환 방식의 병렬처리 컴퓨터용 마이크로 프로세서로써 단일 프로세서내에 병렬처리가 가능하도록 구현되어 있으며 가격 대 성능비가 우수한 시스템으로 강력한 병렬 처리 시스템으로의 구현이 가능한 프로세서이다. 트랜스퓨터의 특징은 하드웨어적인 작업 전환장치, 실수연산 프로세서, 내부메모리, 마이크로프로그램으로 내장된 스케쥴러등을 제공하고 있고, 프로세서 자체내에 4개의 통신링크를 이용하여 여러개의 프로세서를 연결하므로서 강력한 병렬 처리 시스템을 구현할 수 있을 뿐만 아니라, 응용프로그램에 따라 파이프라인 구조, 링 구조, 트리 구조, 메쉬 구조, 하이퍼큐브 구조와 같은 토폴로지의 구현이 가능하며, 메시지의 전달이 자유로운 장점을 이용하여 미사일 유도장치, 로봇공학, FA시스템, 전기통신, 그래픽스 분야등 다양한 응용프로그램에 적용 가능한 프로세서로 많이 사용되고 있다.

시스템의 성능은 다수의 프로세서를 사용하여 작업들을 처리시 각 프로세서에게 자료를 균등하게 할당하는 방법에 따라 성능향상을 가져오는데 이들 작업들은 작업간에 의존 관계나 데이터 분할 방법에 따라서 모든 프로세서가 같은 양의 작업을 수행하도록 한다. 이러한 방법에는 Processor Farm 모델, 파이프라인처리 모델, Data Parallelism 모델, 라우팅 프로그램과 같은 자료 분배 방법이 있다.

Processor Farm 모델은 독립적으로 수행 가능한 작업들을 하나의 Controller와 여러개의 Worker 프로세서를 이용하여 프로세서간의 작업을 균등하게 배분할 수 있는 자료 분산 모델이다. 파이프라인 처리 모델은 모든 데이터들이 모든 프로세서를 통해 전달되고 각 프로세서들은 각 데이터 원소들에 대하여 서로 다른 동작을 수행하게 한다. 이 모델은 프로세서간 특별한 통신 기술을 요구하지 않고 있으므로 자료간 통신과 프로그램 작성이 쉬워 가장 간단한 병렬 시스템을 생성할 수 있다. Data Parallelism 모델은 모든 프로세서에게 완전한 작업들을 주고 데이터를 분할하여 각 프로세서에게 분할된 데이터만을 처리하게 하는 방법으로 프로세서간의 토폴로지를 다양하게 제공할 수 있으나 데이터의 균등한 분할과 프로세서간 자료 교환을 위해 발생되는 통신 오버헤드를 줄이기 어렵고 프로그램의 작성이 복잡한 단점을 가지고 있다. 라우팅 프로그램은 Message-passing Parallel Computer 생성을 위한 프로그램으로 병렬처리 구현에 유용하며 네트워크 구조에서 모든 프로세서가 다른 모든 프로세서로 정보를 보낼 수 있는 통신 구조를 제공하며, 메시지의 송수신 및 계산 작업을 수행하도록 함으로써 Controller와 같은 특정한 프로세서가 필요 없이 프로세서간의 deadlock 문제를 해결한다.

III. 병렬 처리 기법

다수의 프로세서들을 가진 병렬처리 시스템에서 속도 향상이 프로세서 수 만큼 향상되지 못하는 주된 원인은 프로세서간 통신에 따른 오버헤드와 프로세서들에게 균등한 작업량을 할당하지 못하기 때문이며, 데이터의 의존성이 존재할 경우 일부 프로세서들이 대기 상태에 있어 발생되는 낮은 시스템 이용률 때문이다. 이러한 성능 저하를 개선하기 위한 방법으로 균등한 작업 할당, 프로세서간의 통신량, 교환하는 데이터량 최소화, 데이터간 의존성이 있을 경우 하나의 프로세서 또는 인접한 프로세서에게 할당과 같은 사항들이 고려되어야 한다.

자료 분할은 주어진 문제에 대하여 자료나 태스크들 사이의 종속성을 판단한 후 순차처리 할 것인지 병행 처리 할 것인지를 결정하고 이때 분할된 자료나 태스크들은 균등한 작업 분배를 통해 실행 시간을 최소화하고, 시스템의 이용률을 크게 하도록 분할한다. 또한 통신 오버헤드를 최소화하기 위해 프로세서들간의 데이터의 교환량이 적어지도록 하여 프로세서간의 통신량을 최소화 할 수 있도록 분할한다. 이는 분할된 적절한 동기화와 스케쥴링에 의해 과부하를 감소시키므로 효율적인 병렬처리가 가능하다. 프로세서 스케쥴링은 태스크를 처리할 프로세서를 지정하고 처리 순서를 정해 주는 기법으로 각 태스크의 처리에 걸리는 시간과 태스크 간의 처리 순서 및 자료 종속 관계에 대한 정보를 가지고 스케쥴링 한다. 스케쥴링은 프로세서의 수가 증가하면서 처리속도를 향상 시킬 수 있고, 만약 같은 시간 내에 처리가 완료 될 수 있다면 가능한 적은 수의 프로세서를 사용하도록 스케쥴링 하는 것이 시스템의 효율을 향상 시킬 수 있다.

여러 개의 프로세서들이 독립적으로 프로그램을 실행하는 병렬 처리 컴퓨터는 프로세서간의 데이터 교환이 필요하고 자료간 종속성이 있는 문제들을 처리할 경우 병렬 처리를 하기 위해 동기화가 필요하다. 이러한 동기화 기법에는 수행 방법에 따라 임의 동기화, 장벽 동기화, 파이프라인 동기화, 임계 영역 동기화 기법이 있다. 동기화를 위한 위의 기능들은 단일 프로세서에서는 필요하지 않은 기능으로 다수의 프로세서들이 분할된 문제를 동시에 처리하고 시스템 자원을 공유하기 위해 추가되는 기능으로 프로그램 코드 및 수행 시간은 병렬처리를 위한 오버헤드가 발생하는 것을 줄임으로 효과적인 병렬처리 수행이 가능하도록 한다. 또한, 단일 프로세서에서 수행하던 작업을 병렬 처리 시스템에서 실행하기 위해서는 새로운 병렬 알고리즘의 적용이 필요하며, 병렬 알고리즘은 자료간 병렬성을 높이고 프로세서간 통신량을 최소화 하도록 구현 되어야 한다.

프로세서들 간의 연결 형태는 응용프로그램에 따라 파이프라인 구조, 링 구조, 트리 구조, 메쉬 구조, 하이퍼큐브 구조와 같은 토폴로지의 구현이 가능하며, 메시지의 전달이 자유로운 장점을 이용하여 해당 문제에 가장 적합한 프로세서망을 구성한다. 그림 1의 (a), (b)는 각각 프로세서의 수가 최대 10개 있을 경우 링을 이용한 메쉬(Mesh) 구조와 하이퍼큐브 구조의 시스템 토폴로지를 나타내고 있다.

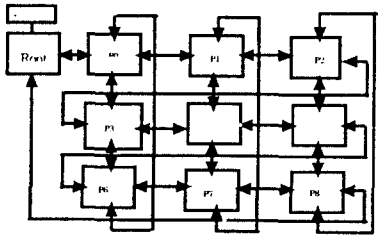
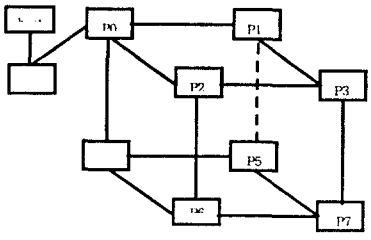


그림 1 (a) 링을 이용한 메쉬 구조



(b) 프로세서 8개를 이용한 하이퍼큐브 구조

그림1 (a)의 메쉬 구조를 이용한 시스템 토폴로지는 모든 프로세서가 4개의 통신링크를 이용하여 다른 프로세서와 연결되고 이 구조에서 직접 연결되지 않은 프로세서들간에는 인접한 프로세서를 경유하여 자료를 교환한다. 이와 같은 구성은 무한대의 프로세서를 이용하여 병렬 시스템을 구성할 경우 링 구조 보다 프로세서간 통신 오버헤드를 줄일 수 있으나 프로그램의 구현이 복잡하고 프로세서의 수가 증가 시 시스템 구성을 재 작업하여야 하는 단점이 있다. 그림 (b)는 3차원 하이퍼큐브(HyperCube) 구조의 형태로 라우팅 함수를 적용시켜 프로세서들간의 관계를 2진수로 표현한 후 프로세서 8개를 이용하여 큐브형태로 연결한다. 이 형태는 프로세서의 연결 상태를 고정시켜 놓고 응용 프로그램에 적용시킨다.

4. 실험 모델 설정 및 방법

재구성 컴파일러 알고리즘에서 추출된 병렬 코드를 이용하여 실제 병렬처리 시스템에서 적용하여 실행시키기 위해서는 자료간의 의존관계에 따른 데이터 분할, 시스템 토폴로지 결정, 자료분산 모델 결정, 스케쥴링 결과를 통한 각각의 프로세서에게 작업 할당하는 방법 결정, 의존관계가 있는 데이터 처리시 고려되는 동기화 방법 결정에 대한 조건들이 먼저 선행되어져야 한다. 다음은 포트란으로 작성되어진 순차 코드와 순차 코드로부터 병렬성이 추출된 병렬 코드의 일부분을 나타낸다. 순차 코드는 순차 처리 시스템에서 수행 가능한 코드이고, 병렬 코드는 하나의 트랜스퓨터로부터 10개의 트랜스퓨터로 수행 가능한 코드이다.

```
DO I = 3, N1                          DO 10 II = 3, N1, 2
  DO J = 5, N2                          DOALL 10 I = II, II + 1
    A(I, J) = B(I-3, J-5)               DOALL 10 J = 5, N2
    B(I, J) = A(I-2, J-4)                 A(I, J) = B(I-3, J-5)
                                          B(I, J) = A(I-2, J-4)
```

(a) 순차 코드(Sequential Code)　　(b) 병렬코드(Parallel Code)

그림 2

위의 그림 1의 (b)에서 I와 J에 대한 DOALL 문장은 병렬 처리가 가능하므로 동시에 처리 되어질 수 있다. 데이터 분할이 가능하도록 순차 및 병렬처리가 가능한 부분을 그림 2을 통해 알아보면 다음과 같다.

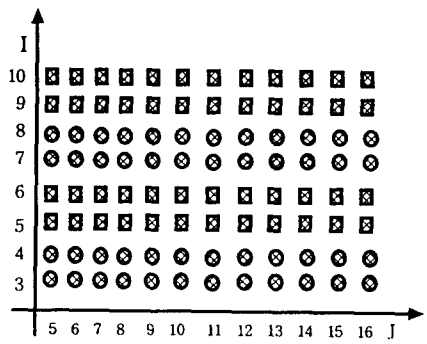


그림 2 자료 종속에 의한 데이터

N1과 N2값을 각각 10, 16이라 하면 동일한 모양의 데이터 원소들은 여러 개의 프로세서에서 병렬처리가 가능하고 I값의 증가에 따른 다른 모양의 데이터 원소들은 순차적으로 처리해야만 한다.

- **자료분할(data partition)** : 작업할 프로세서의 수를 P_num, N1, N2의 크기에 따른 병렬처리가 가능한 원소의 수를 ParJob_size, 각 프로세서에 할당될 Packet Size는 ParJob / P_num한 값이고, 프로세서 수와 N1, N2에 따른 작업 크기에 따른 프로세서 할당되는 방법을 알아본다. 만약 ParJob_size / P_num = 0일 경우는 각 프로세서에 동일한 작업량을 줌으로 써 부하의 불균등을 막고 프로세서들이 거의 비슷한 시간에 종료된다. 따라서 프로세서의 이용률을 증가 시킬 수 있다. 이 때 시스템의 성능 향상은 프로세서의 네트워크 모양에 의존하며 프로세서의 수에 따라 수행 시간이 결정되어진다. 또한 ParJob_size / P_num ≠ 0일 경우는 병렬 처리 할 수 있는 작업을 프로세서들에게 동일한 양으로 할당할 수 없는 경우이다. N2의 값이 100이고 프로세서의 수를 9로 했을 때 22개의 데이터 원소들을 모든

프로세서가 동일하게 수행하고도 2개의 데이터 원소들이 남는다 이 때 다른 프로세서들과 가장 적은 통신을 하는 프로세서에게 이 작업을 할당하여 처리하도록 한다. 따라서 계산 속도와 통신 속도를 고려하여 작업을 분할한다.

· 시스템 토폴로지(system topology) : 링(Ring)

가장 간단하게 시스템 토폴로지를 구성할 수 있으며 첫 번째 프로세서와 마지막 프로세서를 연결시키므로 링 형태가 되도록 구성한다. 본 연구에서 실행하고자 하는 병렬 코드를 링 구조를 이용하여 수행할 경우 프로세서 수를 1~10개를 사용시 프로세서 하나는 루트 프로세서로 호스트와 연결하여 입출력 및 콘솔을 담당하게 하고 나머지 9개의 프로세서로 계산 작업을 처리하도록 한다. 링 구조는 프로세서 수에 따른 수행 속도를 실험 할 때 특별한 기술이 없이도 재구성이 쉽다. p1에서 p4로 자료의 교환은 프로세서 순서에 따라 p2, p3를 경유하여 자료 전송이 이루어진다.

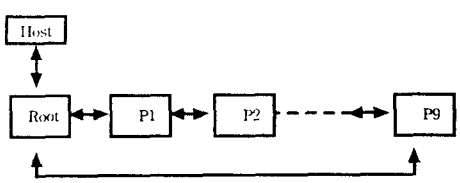


그림 3 트랜스퓨터 10개를 이용한 링 구조

· 병렬 코드 수행을 위한 자료 분산 모델

병렬 처리 시스템의 성능 향상을 위해서 효율적인 자료 분산은 시스템의 이용률을 높일 수 있다.

프로세서 Farm은 Controller가 각 worker 프로세서에게 처리할 작업을 주고 모든 worker 프로세서로 부터의 결과를 모아 다음 처리할 작업을 다시 보내주는 작업을 하고 각각의 worker 프로세서는 controller 프로세서로부터 할당된 작업을 자신의 버퍼에 복사하고 다음 프로세서에게 전달한다. 그리고 할당된 작업을 계산하고 자신이 계산한 결과 값과 다음 프로세서로부터 계산된 결과를 받아 controller는 자료를 처리한다. 프로세서 Farm에 의한 처리 과정을 도식화 하면 그림 7과 같다.

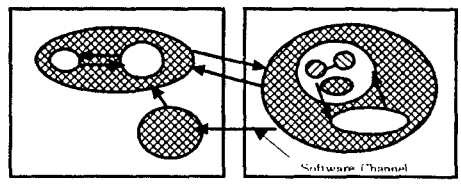


그림 4 프로세서 Farm을 이용한 병렬코드 수행을 프로세스간 자료 처리 과정

5. 실험 결과 및 분석

응용프로그램에서 종속성이 있는 작업들을 효율적인 자료 분배와 소프트웨어적인 여러 가지 기법들을 이용하여 실제 병렬 처리 시스템에서 수행 하였다. 수행되어질 응용 프로그램은 자료간 종속성이 있는 문제로 재구성 컴파일러 알고리즘에서 추출된 병렬 코드로 다수의 트랜스퓨터로 구성된 Mc-3 Machine에서 수행 하였다. 프로세서의 토폴로지는 간단하게 네트워크를 구성할 수 있고 프로세서 증가에 따른 속도 향상을 알아보고자 할 때 프로세서의 추가에 따른 새로운 프로세서 기술이 필요 없이 확장이 용이한 장점을 가진 링 구조를 선택하였다.

프로세서 수는 1~10개의 노드를 이용하였다. 재구성 컴파일러 알고리즘을 이용하여 추출된 병렬 코드는 자료간 종속성이 있는 작업이기 때문에 나누어 처리 되었던 이전 작업의 결과를 모든 프로세서가 수행시 알고 있어야 하는 단점을 극복하기 위해 별도의 버퍼를 두어 이전 결과를 기억 할 수 있도록 하였다. 병렬 코드 수행시 처리될 자료들은 프로세서 Farm 모델 개념에 기초하여 각 프로세서에 자료를 분배 하였다. 프로세서수의 증가에 따른 속도향상은 표 1과 같다. 작업량에 따른 속도 향상을 알아보기 위해 N1, N2의 값을 30, 50, 100에서 수행 하였다. 또한 프로세서 하나를 사용한 때와 2~9를

프로세서 수 / N1, N2	P1	P2	P3	P4	P5	P6	P7	P8	P9
30,30	194	173	169	165	161	160	160	159	159
50,50	-	608	585	574	550	547	543	541	540
100,100	-	1225	1151	1123	1107	1095	1088	1082	1077

표 1 프로세서수의 증가와 가변작업에 따른 실행 시간

사용할 때의 수행 속도를 비교하고 그림 5와 같이 성능 평가 하였다.

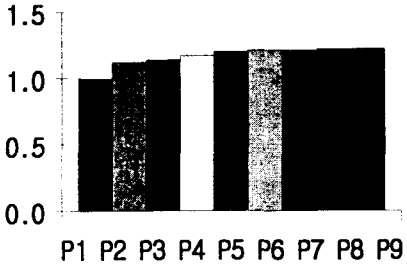


그림 5 프로세서 수의 증가에 따른 성능 비교

효율(E)은 프로세서의 수가 n이고 n개를 이용하여 걸린 수행 시간을 Tn이라 하면 식 1과 같이 정의할 수 있다.

$$\text{Efficiency}(E) = \frac{T1}{Tn \times n} \times 100(\%)$$

(식1)

그림 6 보는 바와 같이 프로세서의 수를 증가시에 따른 효율성을 보면 프로세서 2개를 사용하여 수행했을 때 56%로 가장 좋고, 프로세서 5개 이상을 사용 했을 때는 효율성이 감소 되는 것을 알 수 있다.

이는 자료간 종속이 존재하여 프로세서의 수가 증가할 때 프로세서간 통신량이 많아짐에 따라 속도가 저하됨을 알수 있고 링 구조에 따른 프로세서간 통신 시간이 오래 걸리기 때문이다

6. 결론

본 논문에서는 병렬처리시 성능향상을 줄 수 있는 적당한 토폴로지를 구성하고 응용프로그램에 적합한 토폴로지를 결정하는 문제에 대하여 알아 보고 병렬

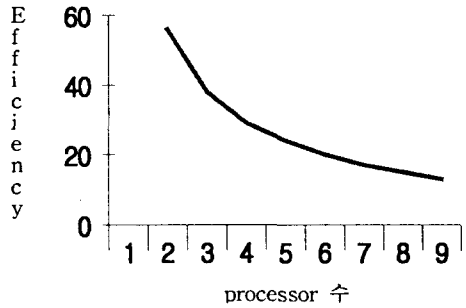


그림 6 프로세서 수에 따른 효율성

성을 극대화 시키고 시스템의 효율을 높일 수 있는 자료분산, 데이터분할, 동기화, 스케쥴링, 병렬알고리즘과 같은 소프트웨어적인 기법을 알아 본 후 프로세서의 수를 증가하며 동일한 작업을 수행한 결과 프로세서의 수가 2개 일 때 Cost-efficiency임을 알 수 있다. 또한 프로세서의 수를 6개 이상 사용할 때는 처리 속도의 증가가 거의 없음을 알 수 있다.

그 이유는 자료간 종속이 있는 처리된 자료에 대하여 프로세서간 통신량이 많아졌기 때문이다. 이러한 문제 해결을 위해 각 node의 토폴로지, 자료분산 모델, 프로세서수를 고려하여 수행하므로 더 많은 성능 향상을 기대하고 추후 트랜스퓨터의 사용을 용이하게 하고자 트랜스퓨터에 적용시킬 수 있는 소프트웨어적인 많은 문제들과 적용 방법들에 대하여 연구하고자 한다.

참고 문헌

[1] D.B.Loveman, "High Performance Fortran," IEEE Parallel & Disrtibuted Technology, Vol. 1, No. 1, pp. 25-42, Feb. 1993.

[2] 박두순, 이광영, 황종선, 김병수, "병렬처리를 위한 동기화 기법", 정보과학회지(제13권, 제7호),

[3] M. E. Wolf, "Multiprocessor Synchronization for Concurrent Loops", IEEE Software, 1988.

[4] 김종현, "병렬처리 구조론" , 생능출판사 pp188

[5] Ronald S.Cok, "Parallel programs for the Transputer", Cok