14주차 1차시 병렬알고리즘 I

【학습목표】

1. 최소값 찾기을 이해할 수 있다.
2. 리스트 순위 부여를 이해할 수 있다.

학습내용1 : 기본 개념

⦿ 병렬 컴퓨터에서 수행되도록 작성된 알고리즘

■ 병렬 컴퓨터
• 복수 개의 프로세서를 사용하여 여러 개의 명령을 동시에 수행할 수 있는 컴퓨터
• 다양한 종류의 병렬 컴퓨터가 개발/사용
- 단일 시스템으로 구성 → 듀얼 코어, 쿼드 코어와 같이 하나의 CPU 내에 복수 개의 코어를 넣어 여러 명령어를 동시에 수행
- 여러 시스템으로 구성
• 병렬 OS, 병렬 PL, 병렬 알고리즘 등이 필수적

1. 병렬 컴퓨터의 분류

⦿ 분류 기준
■ 프로세서들이 메모리를 공유하느냐? 메모리가 분산되어 각 프로세서에 속하여 있는가?
• 공유 메모리 shared memory 모델
• 연결망 interconnection network 모델
▪ 프로세서들이 어떠한 형태로 연결되어 있는가?
▪ 모든 프로세서가 특정 순간에 동일한 명령을 수행하여야 하는가? 아니면 서로 다른 명령을 수행해도 되는가?

⦿ Flynn에 의한 병렬 컴퓨터분류실제로 구현된 적이 거의 없으며 이론적으로 제시되고 있다.MISD

구분	내용
SISD	Single Instruction stream, Single Data stream a. 일반적인 컴퓨터 구조(본 노이만 방식) b. 단일 명령어 흐름에 따른 단일 데이터 흐름 컴퓨터 c. 하나의 명령에 따른 하나의 자료를 처리하는 구조를 가진다. d. 병렬 처리는 파이프라인(Pipeline) 구조로 구현한다.
SIMD	Single Instruction stream, Multiple Data stream a. 단일 명령어 흐름에 따른 다중 데이터 흐름 컴퓨터 b. 하나의 명령에 따른 여러 자료를 동시에 처리하는 구조를 가진다. c. 병렬 처리는 배열(Array) 처리와 벡터(Vector) 처리 구조로 구현한다. d 모든 데이터의 요소들을 같은 계산으로 수행
MISD	Multiple Instruction stream, Single Data stream a. 다중 명령어 흐름에 따른 단일 데이터 흐름 컴퓨터 b. 여러 명령에 따른 하나의 자료를 처리하는 구조를 의미한다. c. 여러 명령이 하나의 자료를 동시에 처리할 경우 작업의 일관성이 무너질 수 있다. d. 실제로 구현된 적이 거의 없으며 이론적으로 제시되고 있다. 다수 명령어 스트림, 단일 데이터 스트림 시스템
MIMD	Multiple Instruction stream, Multiple Data stream a. 다중 명령어 흐름에 따른 다중 데이터 흐름 컴퓨터 b. 여러 명령에 따른 여러 자료를 동시에 처리하는 구조를 가진다. c. 여러 개의 처리기가 독립적인 명령어로 각각 다른 자료를 처리. d. 병렬 처리는 다중 처리(Multi Processing) 구조로 구현한다. e. 처리기 상호연결 시 Tightly Coupled System을 다중처리기, Loosely Coupled System을 분산 처리 시스템이라 함

⦿ 병렬 컴퓨터의 이론적인 모델

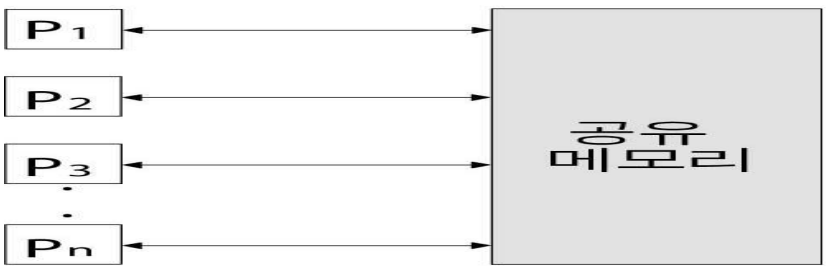
■ 메모리의 형태, 동시 처리 명령어의 형태, 프로세서 간 통신 형태 등

• PRAM, PMH, BSP, LogP 등

⦿ PRAM 모델 (Parallel Random Access Machine)

■ 이상적인 이론적 모델

■ 동기화된 공유 메모리를 가진 MIMD 컴퓨터



2. PRAM 모델에 대한 가정

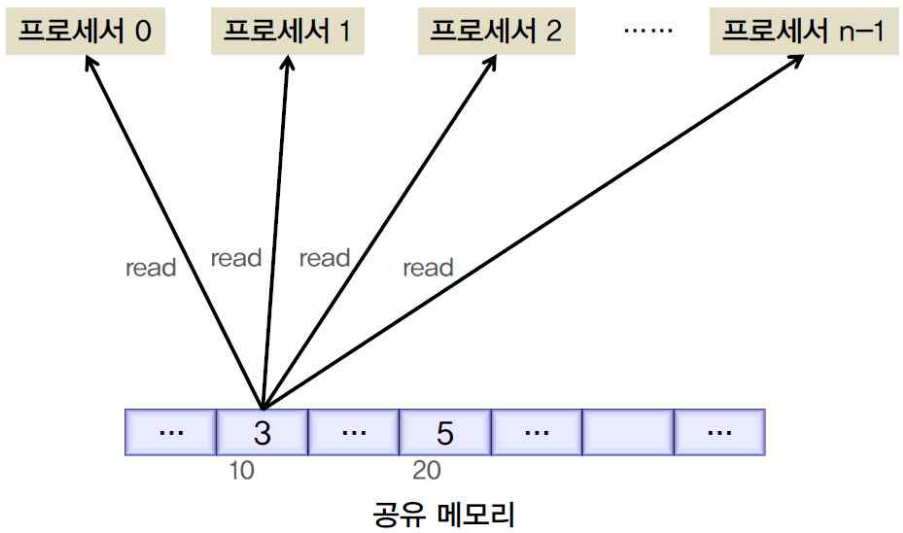
⦿ 프로세서 인덱스 → 1, 2, …, n
■ 각 프로세서는 자신에게 할당된 번호인 인덱스를 알고 있으며 이를 계산에 사용 가능

⦿ 산술/논리 연산을 상수 시간에 수행
■ 특정 순간에 모든 프로세서는 동일한 명령을 수행, 즉 PRAM을 SIMD 모델로 사용

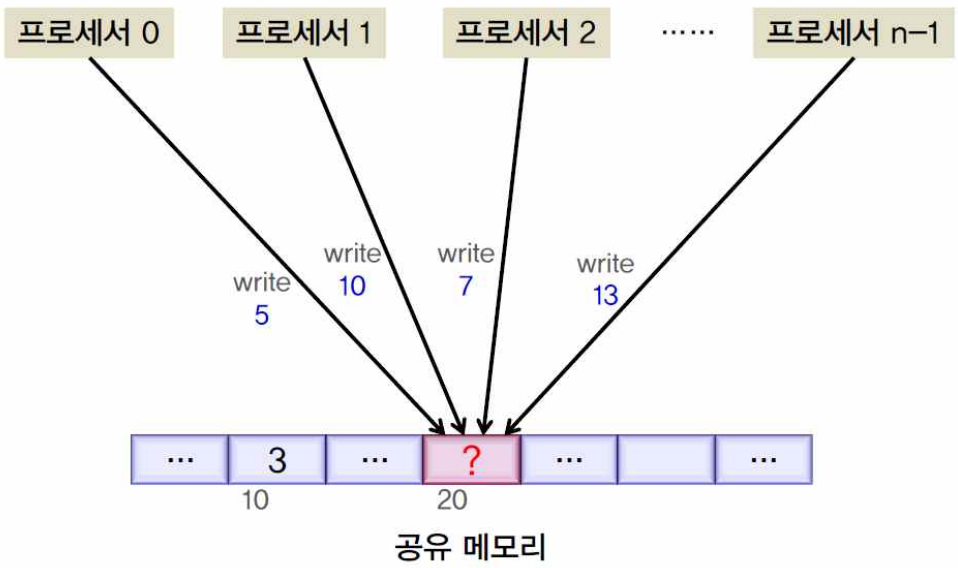
⦿ 모든 프로세서가 주어진 명령을 수행하여야 하는 것은 아니다.
■ 각 프로세서는 자신이 특정 조건을 만족하는가를 확인한 후에 만족하는 경우에만 그 명령을 수행

⦿ 각 프로세서는 메모리의 임의 접근이 가능하며, 읽고 쓰는 데 드는 시간은 상수 시간이다.

* PRAM 모델에서의 읽기



* PRAM 모델에서의 쓰기



⊙ PRAM 모델의 종류

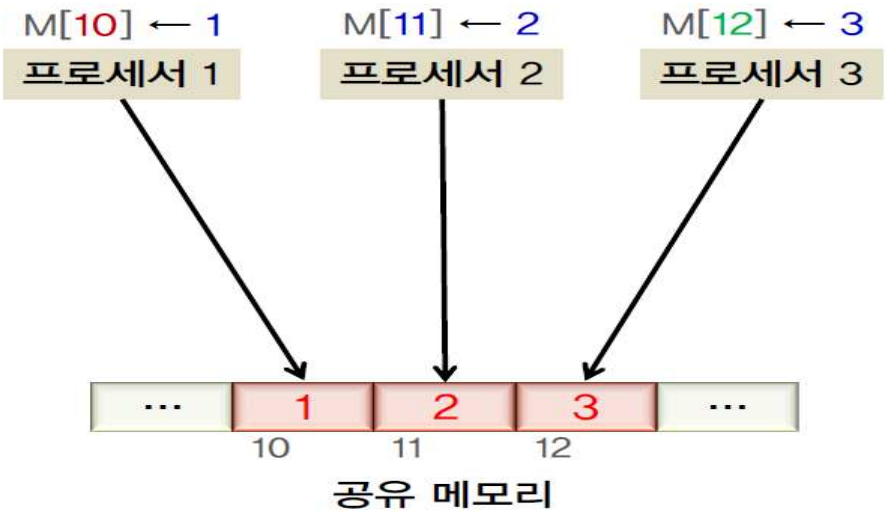
PRAM 모델은 여러 개의 프로세서들이 동시에 메모리에 접근 할 때 그들 중 동시에 어떤 프로세서의 자료가 메모리에 읽고 쓰기가 가능한지 하는 메모리의 접근방식에 따라 다음 4가지의 종류로 나누어지게 된다.

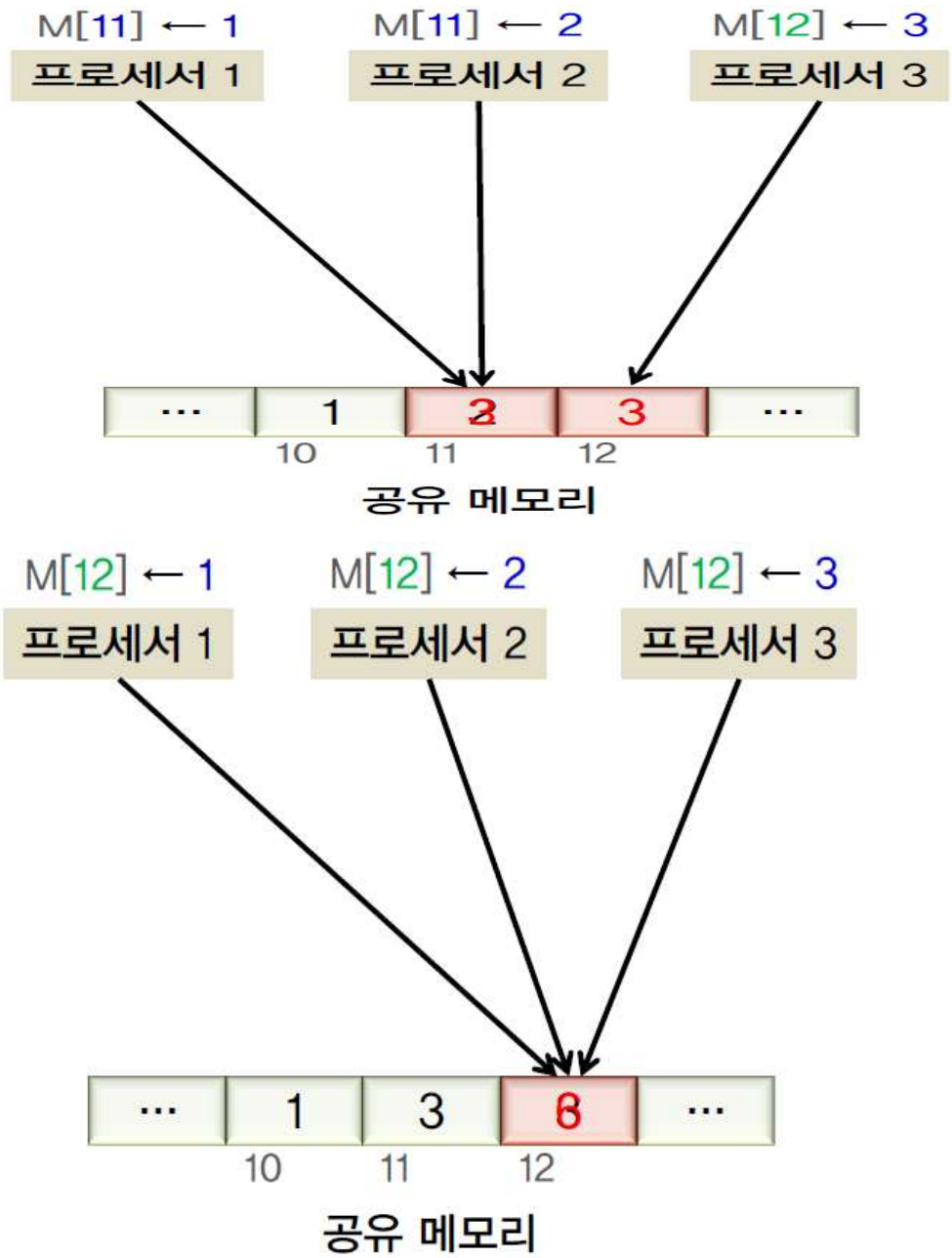
읽고 쓰기 알고리즘	의미
CRCW Concurrent Read Concurrent write	동시 읽기/ 동시쓰기 가장 이상적인(강력한) 방식
CREW Concurrent Read Exclusive write	동시 읽기/ 독점쓰기 가장 현실적인 방식
ERCW Exclusive Read Concurrent write	독점 읽기/ 동시쓰기 현실성이 없는 모델
EREW Exclusive Read Exclusive write	독점 읽기/ 독점 쓰기 가장 제한적인 방식

⊙ CRCW, ERCW의 경우 추가적인 약속이 필요

■ 동시에 메모리에 쓰려고 할 때 어떤 값을 쓸 것인가?
■ 쓰려는 모든 값이 동일한 경우에만 쓰기
■ 쓰려는 값들 중 임의의 한 값을 쓰기
■ 쓰려는 값들 중 최소값을 쓰기
■ 쓰려는 값을 모드 더한 값을 쓰기
■ 기타 미리 정해 둔 우선순위에 따라 쓰기
- 프로세서 중에서 가장 작은 인덱스를 갖는 프로세서의 값 쓰기

CRCW PRAM 모델 동시 쓰기 → 모든 값을 더한 값을 쓰기





3. PRAM 모델의 실용성

⊙ PRAM은 실용적인 모델인가?

- ■ 현재의 기술로는 구현하기 힘든 모델
- ■ 각 셀에 대한 상수 시간 접근을 위해 요구되는 구현의 복잡성을 무시
- ■ 병렬 알고리즘의 개발 분석에 좋은 모델

⦿ PRAM 모델을 사용하는 이유
- ■ 간단하고 수월하게 병렬 알고리즘의 고안이 가능
- ■ 프로세서 간의 통신비용을 무시한 채로 풀고자 하는 문제를 최대한으로 병렬화하는 데에만 초점을 맞출 수 있음
- ■ 고안된 알고리즘의 간단한 변형을 통해 다른 많이 사용되는 병렬 컴퓨터 모델에서 수행 가능
- ■ 다른 특정 모델에 비해 시간/공간/프로세서 수의 복잡도를 분석하는 것이 수월

4. 병렬 알고리즘의 복잡도

⦿ 순차 알고리즘의 경우
- ■ 다항식 시간 → 실용적
- ■ 지수 시간 → 비실용적

⦿ 병렬 알고리즘의 경우
- ■ 입력 자료 개수의 다항식 배만큼의 프로세서를 이용하여 폴리로그 시간 $O(\log^k n)$이 걸리면 실용적이다.

5. 병렬 알고리즘의 효율성 판단 척도

⦿ 시간 효율성, 속도 향상률, 작업량

S(n): 문제 크기 n에 대한 최선의 순차 알고리즘의 수행시간

P(n): 문제 크기 n에 대한 p개의 프로세서를 사용한 병렬 알고리즘의 수행시간

⦿ 시간 효율성

$$TE = \frac{S(n)}{p \times P(n)}$$

이 값은 0과 1사이의 수로서 1에 가까울수록 더 효율적인 알고리즘이 된다.
예를 들어 S(n)=16초 이고 프로세서 4개를 사용하여 4초가 걸렸다고 가정한다면 시간의효율성은 1이 되고 가장 완벽한 최적의 알고리즘이다. 만일 5초가 걸렸다면 16/(4*5) = 0.8이 된다.

⦿ 속도 향상률

$$Sd = \frac{S(n)}{P(n)}$$

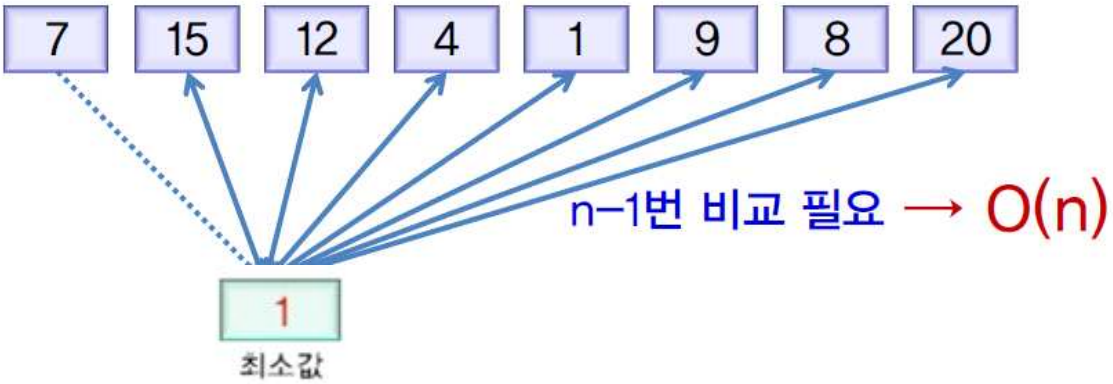
S(n)=16초, P(n)=5, p=4 라면 4개의 프로세서를 사용한다고 해서 4배의 향상이 되는 것이 아니라 16/5=3.2가 되어서 3.2배 향상됨을 볼 수 있다. 또 p=사용프로세서, P(n)=알고리즘의 실행시간으로 p*P(n)이 그 알고리즘의 작업량이 됨을 알 수 있다.

⦿ 작업량
- ■ p×P(n)
- ■ 작업량 = S(n)이면 최적의 병렬 알고리즘

학습내용2 : 최소값 찾기

1. 최소값 찾기(CREW PRAM)

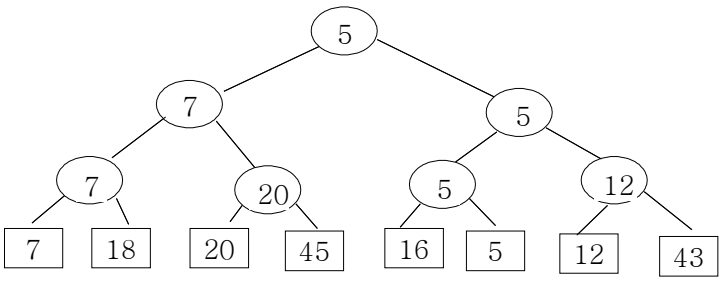
⦿ 최소값 찾는 순차 알고리즘



■ 병렬화 하기 어려운 알고리즘

• i번째 비교가 i-1번째 비교의 결과를 이용하므로 알고리즘의 실행이 매우 순차적

⦿ 이진트리로 표현



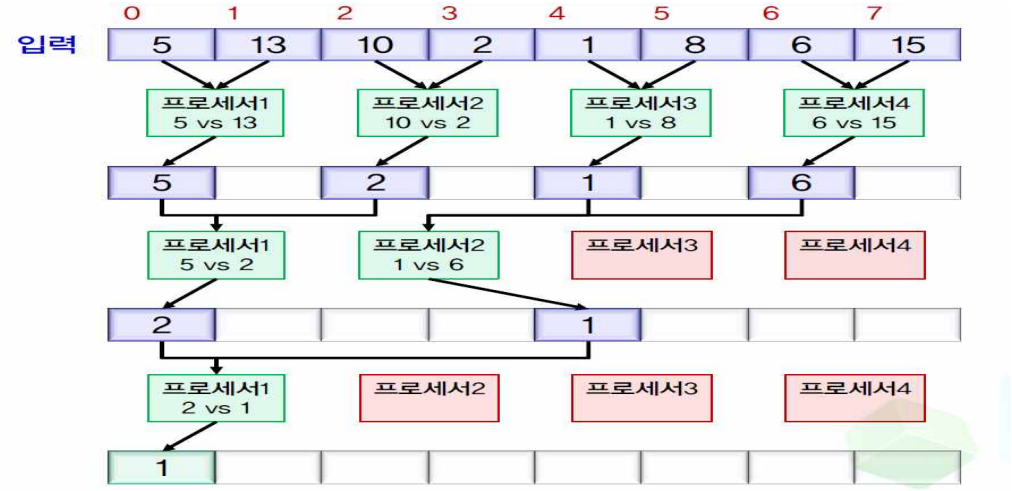
■ 비교회수 n-1회

⦿ CREW PRAM 알고리즘

⦿ 균형 이진 트리 방식, 토너먼트 방식

■ 두 원소씩 골라서 비교 후 작은 값을 선택하여 다음 단계로 보내는 방식

⦿ n/2개 프로세서 → O(logn) 시간



⦿ 작업량

■ 순차 알고리즘 → O(n)

■ 병렬 알고리즘 → O(logn)×n/2

→ 병렬 알고리즘의 성능이 순차 알고리즘에 비해 비효율적

⦿ 시간 효율성

$$\frac{S(n)}{p \times P(n)} = \frac{O(n)}{\frac{n}{2} O(\log n)} = O(\frac{1}{\log n})$$

⦿ 알고리즘이 진행되면 참여하지 않는 프로세서의 수가 점차로 증가하기 때문에 효율성이 떨어짐

2. 최적화된 CREW PRAM 알고리즘

⦿ n/logn개의 프로세서 → O(logn) 시간

⦿ 작업량

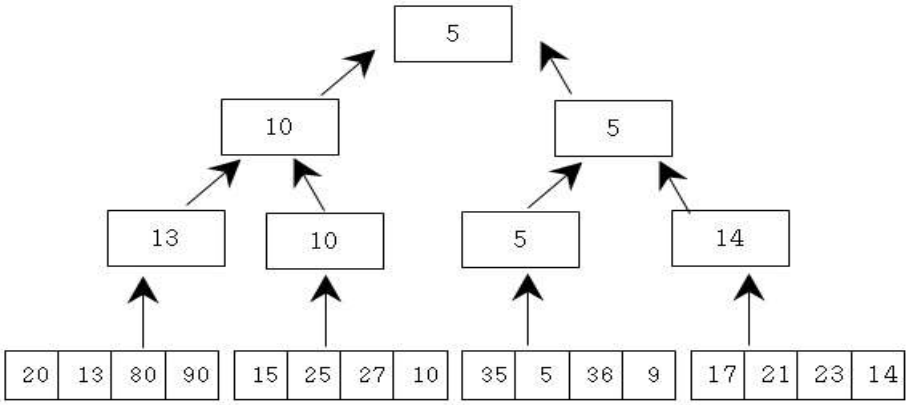
■ O(logn)× n/logn = O(n)

■ 순차 알고리즘 O(n)가 같아서 최적 알고리즘이 됨

⦿ 블록화 blocking

▪동일한 실행시간을 유지하면서 프로세서의 수를 줄여서 최적 알고리즘을 얻는 방식

* 4개의 프로세서를 사용하여 16개의 숫자 중에서 최소값을 구하는 예



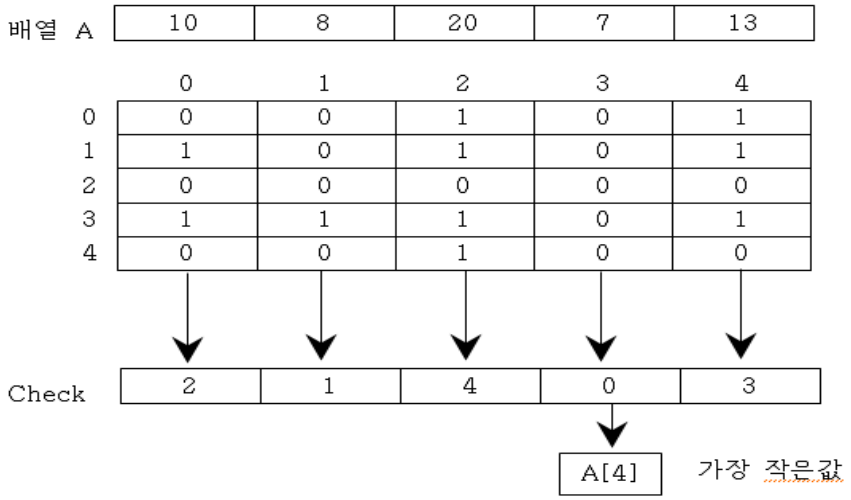
① 제 1단계에서 각 프로세서가 지신에게 할당된 log n개의 데이터로부터 최소값을 순차적으로 구하는 것은 O(log n) 이 된다.
② 두 번째 단계에서 각 블록의 최소값들인 n/log n개의 데이터로부터 최소값을 찾기 위 해 O(log n)의 작업량이 필요하다. n/log n개의 프로세서를 사용하여 O(log n) 시간에 실행을 끝낼 수 있다.
프로세서의 수와 실행시간을 곱한 값이O(n) 이 되어 최적알고리즘이 된다. 이러한 방식으로 프로세서를 줄여서 최적 알고리즘을 구하는 방식을 블록화라 한다.

3. CRCW PRAM 알고리즘

CRCW PRAM 모델은 CREW PRAM 모델과 비교해볼 때 더욱더 강력한 병렬 컴퓨터 모델이다. CRCW PRAM 모델을 사용하면 최소값을 [그림 8-5]와 같이 구할 수 있다.
여기서는 자료가 한 셀에 동시에 쓰여 질 때 그들의 합이 쓰여 진다고 가정한다.
배열 Check[0:n-1]을 사용하고 n^2개의 프로세서를 사용한다. 편의상 프로세서들은 PE(i,j), $0 \le i, j < n$ 으로 번호를 매기기로 한다. Check[i]는 0으로 초기화 한다.

```
FindMini(A, n, Mini)
{
 Check[i] = 0, {PE{(i, 0), 0 ≤ i < n};
 if (A(i) < A(j) or (A[i] = A[j] and i < j)) Check[j] = 1, {PE(i, j), 0 ≤ i, j < n}
 if (Check[i] = 0) Mini = A[i], {PE(i, 0), 0 ≤ i < n};
 }
```

[합을 쓰는 CRCW PRAM을 이용한 5개의 숫자 중에서 최소값을 찾는 예]



⦿ CREW PRAM 모델보다 더 강력한 모델 : CRCW PRAM

⦿ n^2개 프로세서 → O(1)

■ n개의 프로세서

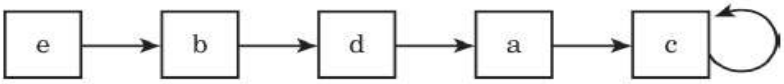
→ 한 원소를 동시에 모든 n개의 원소와 비교 가능

■ n^2개의 프로세서

→ 모든 원소를 동시에 모든 n개의 원소와 비교 가능

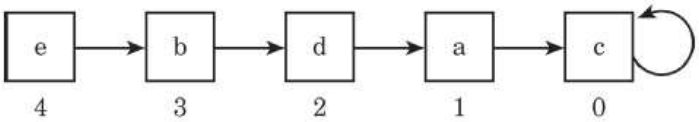
학습내용3 : 리스트 순위 부여

n개의 수로 만들어진 리스트가 다음과 같이 주어졌다고 하자 n의수는 배열 A[0]에서 배열 A[n-1]에 있고 a[i]의 다음에 오는 수의 배열 인덱스가 P[i]에 들어 있는 것으로 가정한다. [아래 그림]에 5개의 수를 갖는 리스트가 있다. Head는 리스트의 첫 번째 원소가 있는 인덱스이다.



	0	1	2	3	4
배열 A	a	b	c	d	e
인덱스 P	2	3	2	0	1

인덱스 P 배열의 P[0]=2는 a는 c를 가르치고 있어서 배열 A에서 c의 순서 2를 의미 한다.
인덱스 P 배열의 P[1]=3는 b는 d를 가르치고 있어서 배열 A에서 d의 순서 3를 의미 한다.
인덱스 P 배열의 P[2]=2는 c는 자신을 가르치고 있어서 배열 A에서 c의 순서 2를 의미 한다.
인덱스 P 배열의 P[3]=0는 d는 a를 가르치고 있어서 배열 A에서 a의 순서 0를 의미 한다.
인덱스 P 배열의 P[4]=1는 e는 b를 가르치고 있어서 배열 A에서 b의 순서 1를 의미 한다.



Head

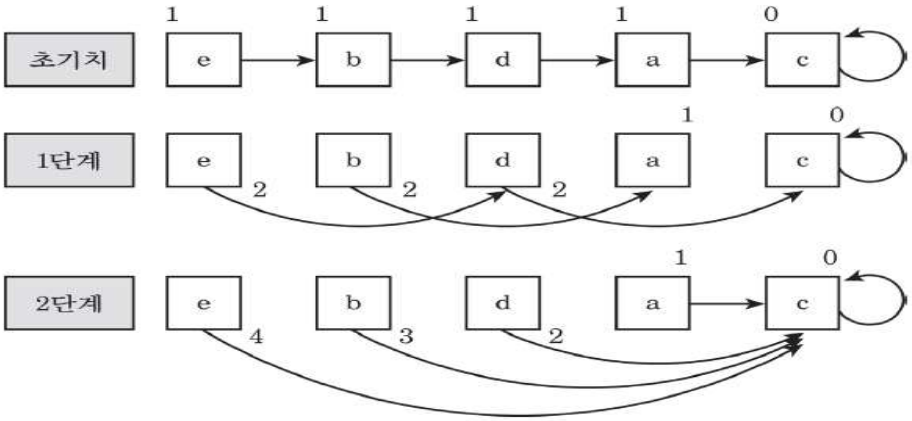
2

	0	1	2	3	4
순위 R	1	3	0	2	4

* 출력 데이터

R[2]=0은 A[2]인 c를 의미하며 0번째임을 의미한다.
좀 더 알고리즘을 자세히 살펴보면 다음과 같다.
R[0]=1은 A[0]인 a를 의미하며 1번째 위치임을 나타낸다.
R[1]=3은 A[1]인 b를 의미하며 3번째 위치임을 나타낸다.
R[3]=2은 A[3]인 d를 의미하며 2번째 위치임을 나타낸다.
R[4]=4은 A[4]인 e를 의미하며 4번째 위치임을 나타낸다.

* 리스트에 순위를 부여하는 CREW PRAM 알고리즘
[아래그림] 입력 데이터와 인덱스를 보면 A[i]인 각 원소들이 A[P[i]]를 가리키고 있으며 두 원소 사이의 거리는 1임을 알 수 있다. 이를 R[i]에 초기치로 저장하고 원소사이의 거리는 R[i]에 저장되며 이 값이 자신의 순위가 된다. 알고리즘은 logn 단계로 구성된다.
이와 같이 거리가 1인 원소로부터 거리가 2인 원소로, 또 거리가 4인 원소로 포인터를 옮기면서 해결하는 기법을 포인트 점핑(Pointer Jumping)다른 말로 더블링(doubling)이라 한다.



- 포인트 점핑을 이용하여 순위를 부여

【학습정리】

1. 병렬 알고리즘
- 병렬 컴퓨터에서 수행되도록 작성된 알고리즘
→ 병렬 컴퓨터 : 복수 개의 프로세서를 사용하여 여러 개의 명령을 동시에 처리할 수 있는 컴퓨터

* PRAM(Parallel Random Access Machine)
- p개의 프로세서와 모든 프로세서가 동등하게 접근할 수 있는 무제한의 공유 메모리로 구성된 병렬 계산 모델

- 메모리 접근 방식에 두는 제약에 따른 분류
 → CRCW: 여러 프로세서가 동시에 같은 메모리 위치를 읽거나 쓰거나 할 수 있다.
 → CREW: 여러 프로세서가 동시에 같은 메모리 위치를 읽을 수 있으나, 동시에 쓸 수는 없다.
 → ERCW: 여러 프로세서가 동시에 같은 메모리 위치를 읽을 수는 없지만 동시에 쓸 수는 있다.
 → EREW: 여러 프로세서가 동시에 같은 메모리 위치를 읽거나 쓰거나 할 수 없다.

2. 최소값 찾기

⦿ CREW PRAM 알고리즘

⦿ 균형 이진 트리 방식, 토너먼트 방식

■ 두 원소씩 골라서 비교 후 작은 값을 선택하여 다음 단계로 보내는 방식

⦿ n/2개 프로세서 → O(logn) 시간

⦿ CREW PRAM 모델보다 더 강력한 모델 : CRCW PRAM

3. 리스트 순위 부여

n개의 수로 만들어진 리스트가 다음과 같이 주어졌다고 하자 n의수는 배열 A[0]에서 배열 A[n-1]에 있고 a[i]의 다음에 오는 수의 배열 인덱스가 P[i]에 들어 있는 것으로 가정한다.