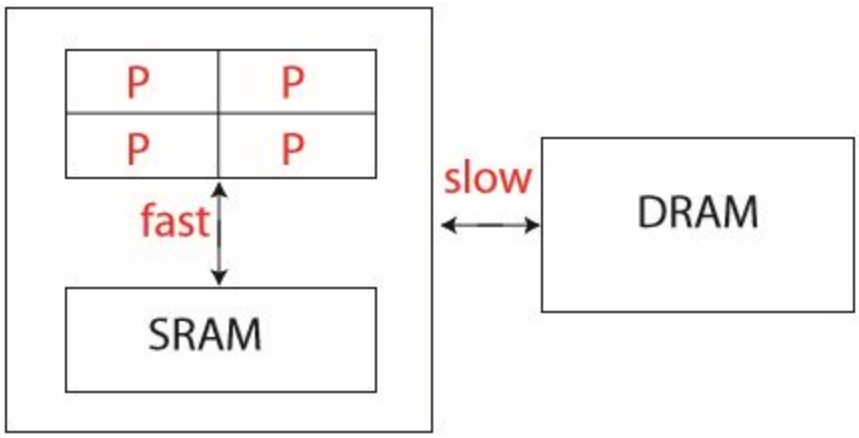
강의 24: 병렬 프로세서 구조 & 알고리즘

프로세서 구조

컴퓨터 구조의 발전:

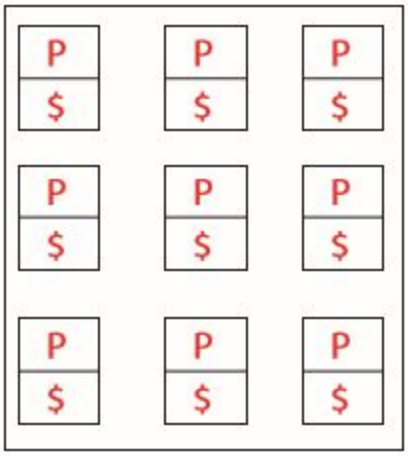
- 인텔 8086 (1981): 5 MHz (IBM PC에 처음 사용됨)
- 인텔 80486 (1989): 25 MHz (숫자를 상표로 사용할 수 없다는 법원 판결 때문에 i486이 됨)
- 펜티엄 (1993): 66 MHz
- 펜티엄 4 (2000): 1.5 GHz (약 30단계의 깊은 파이프라인)
- 펜티엄 D (2005): 3.2 GHz (이후 클럭 속도의 증가가 멈춤)
- 쿼드코어 제온 (2008): 3 GHz (하나의 칩의 코어 수가 성능의 주요 요소)

프로세서는 계산할 데이터를 필요로 함:

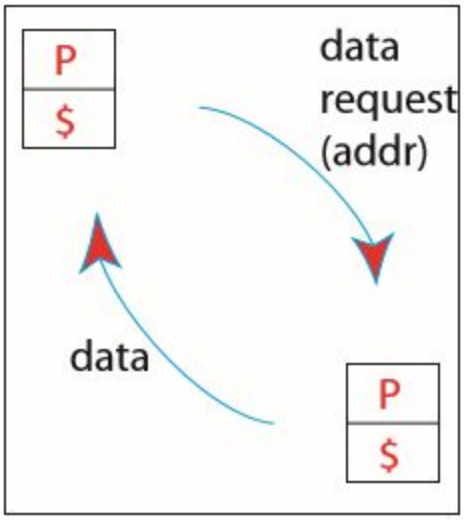


문제점: SRAM은 4개보다 많은 메모리 요청을 병렬적으로 처리 불가

$: 캐시 P: 프로세서



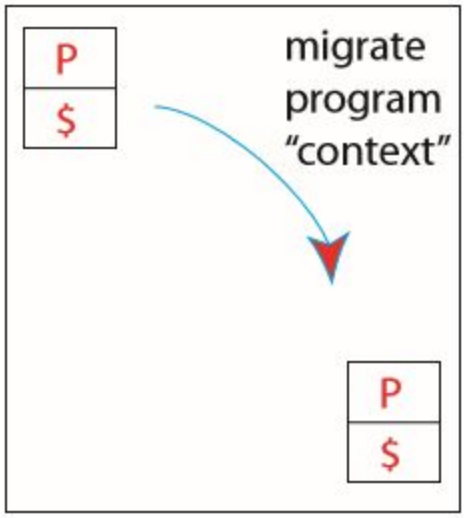
프로그램이 실행되는 대부분 동안 프로세서는 로컬 또는 “캐시” 메모리에 접근
때때로 원격 메모리에 접근:



데이터를 얻기 위해 왕복 접근 필요

연구 주제: 실행 이동

어떤 프로세서에서 실행되는 프로그램이 다른 프로세서의 캐시 메모리에 접근을 필요로 하면, 프로세서는 자신의 "문맥"을 원격 프로세서로 이동시킨 뒤 거기에서 실행된다:



편도 데이터 접근

$$\text{Context} = \underbrace{\text{ProgramCounter} + \text{RegisterFile} + \ldots}_{\text{fewKbits}}$$

(접근되는 데이터보다 클 수 있음)

프로그램의 접근 패턴을 알거나 예측할 수 있다고 가정하자

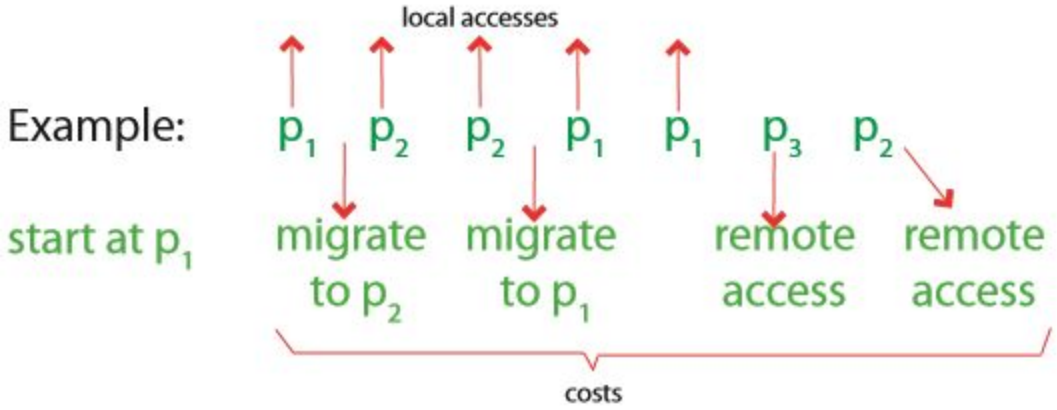
$m_1, m_2, \ldots, m_N$ (메모리 주소)
$p(m_1), p(m_2), \ldots p(m_N)$ (각 mi의 프로세서 캐시)

예시

$p_1 p_2 p_2 p_1 p_3 p_2$
이동비용(s, d) = 거리$(s, d) + L$ ← 적재 대기시간 L은 문맥 크기의 함수
접근비용$(s, d) = 2 *$ 거리(s, d)
만약 $s == d$이면, 비용은 0으로 정의

문제

총 메모리 비용을 최소화하는 이동 시점을 결정하라. 예를 들어:



이 문제를 풀기 위해 무엇을 사용하면 되는가?
동적계획법!

동적계획법 해법

프로그램의 최초 위치 p, 프로세서의 수 = Q

하위 문제?

DP (k, pi): 프로그램이 p1에서 시작해서 pi에서 끝나는 경우 m1에서 mk까지의 메모리 접근의 prefix에 대한 최적해의 비용으로 정의

$$DP(k+1,p_j) = \begin{cases} DP(k,p_j) + \text{cost}_{\text{access}}(\text{p}_\text{j}, \text{p}(\text{m}_{\text{k+1}})) & \text{if } p_j \neq p(m_{k+1}) \\ MIN_{i=1}^{Q}(DP(k,p_i) + \text{cost}_{\text{mig}}(\text{p}_\text{i}, \text{p}_\text{j})) & \text{if } p_j = p(m_{k+1}) \end{cases}$$

복잡도?

$$O(\underbrace{N \cdot Q}_{\text{no.of subproblems}} \cdot \underbrace{Q}_{\text{cost per subproblem}}) = O(NQ^2)$$

제 연구 그룹은 이 분석에 기반한 이동 예측기를 사용하는 실행 이동 머신이라 불리는 128 코어 머신을 설계하고 있습니다.

강의 24: 연구분야들과 6.006 이후의 것들

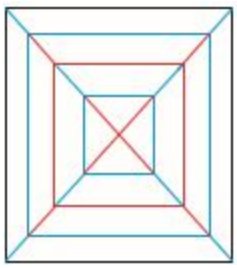
Erik의 주요 연구분야

- 계산 기하학 [6.850]
  - 기하학적 접기 알고리즘 [6.849]
  - 자가 조립
- 자료구조 [6.851]
- 그래프 알고리즘 [6.899]
- 레크리에이션 알고리즘 [SP.268]
- 알고리즘적 조형물

기하학적 접기 알고리즘: [6.849], 온라인 영상

두 가지 측면: 디자인과 접기 가능성

- 디자인: 사각형 종이를 통해 어떤 다면체던 접는 알고리즘 [Demaine, Demaine, Mitchell (2000); Demaine & Tachi (2011)]
  - 양면 종이 ⇒ 2색 표면 가능
  - 열린 문제: "척도 계수"를 어떻게 가장 최적화 시키는가
  - 예) 최선의 $n \times n$ 체커보드 접기 — 최근에 $\approx n/2 \to\approx n/4$ 로 향상됨
- 접기 가능성: 접음선이 주어졌을 때, 평평하게 접을 수 있는가
  - 일반적으로 NP-완전 Bern & Hayes (1996)
  - 열린 문제: $m \times n$ 산/계곡으로 이루어진 접음선 변환 [Edmonds (1997)]
  - 해결됨: $2 \times n$ Demaine, Liu, Morgan (2011)
  - 쌍곡 포물선 [Bauhaus (1929)] 존재하지 않음 [Demaine, Demaine, Hart, Price, Tachi (2009)]



  - 원형 접음선의 이해

- 모든 직선 그래프는 평평하게 접은 뒤 직선으로 한 번 자르는 것으로 만들어질 수 있음 [Demaine, Demaine, Lubiw (1998); Bern, Demaine, Eppstein, Hayes (1999)]

자가-조립

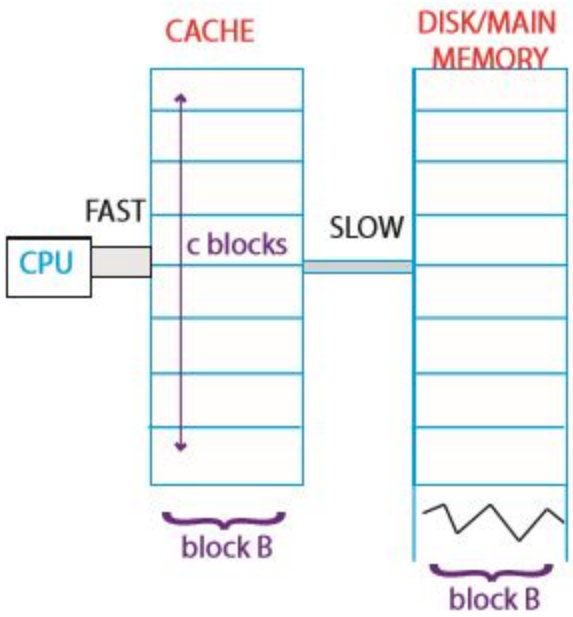
계산의 기하학적 모델

- 접착성 e.g. DNA 가닥, 각 쌍 사이에 힘이 작용
- 양면에 접착성이 있는 사각 타일
- 브라운 운동: 타일/건축 — ∑접착 강도 ≥ 온도 인 경우에 붙어 있음
- O $\left(\frac{lgn}{lglgn}\right)$ 타일 [Rothemund & Winfree 2000] 또는 O(1) 타일 & O(lg n) "단계" algorithmic steps by the bioengineer [Demaine, Demaine, Fekete, Ishaque, Rafalin, Schweller, Souvaine (2007)] 를 사용해서 n × n 사각형을 만들 수 있음
- O(1) 타일과 O(1) 단계를 통해 주어진 알려지지 않은 모형의 ∞ 복제본을 만들 수 있음 [Abel, Benbernou, Damian, Demaine, Flatland, Kominers, Schweller (2010)]

자료구조: [6.851], 다음학기에 영상 제작 예정

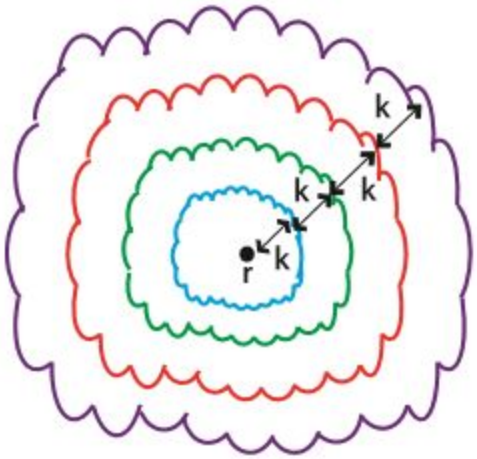
자료구조의 2가지 주요 분류

- 정수 자료구조: 삽입, 삭제, 선행자, 후속자를 조건으로 n개의 정수를 {0, 1, ··· u – 1} 에 저장 (단어 램 위에)
  - 해싱은 O(1) 에 탐색
  - AVL 트리는 모든 것을 O(lg n) 에 수행
  - O(lglg u)/op van Emde Boas
  - O $\left(\frac{lgn}{lglgn}\right)$ /op 퓨전 트리: Fredman & Willard
  - O $\left(\sqrt{\frac{lgn}{lglgn}}\right)$ /op min of above
- 캐시-효율적 자료구조
  - 메모리 전송은 블럭 단위로 일어남 (캐시에서 디스크/메인메모리로)
  - 탐색: $\Theta\left(log_B N\right)$ 전송 (vs. lg n)
  - 정렬: $\Theta\left(\frac{N}{B} log_B \frac{N}{B}\right)$ 전송
  - B & C를 모르는 경우에도 가능 !



(거의) 평면 그래프: [6.889], 온라인 영상

- 다익스트라: O(n) 시간 [Henzinger, Klein, Rao, Subramanian (1997)]
- 벨만-포드: $O\left(\frac{n lg^2 n}{lg lg\, n}\right)$ 시간 [Mozes & Wolff-Nilson (2010)]
- 평면 그래프 위에서도 많은 문제들은 NP-난해이지만, $1 + \varepsilon$ 에 해답을 찾을 수 있음



최적 요소, for any ∈ [Baker 1994 & Others]:

- 어떤 루트 정점 r에서의 BFS 실행
- 모든 k개의 층 삭제
- 많은 문제들에 대해서, 해답은 $1 + \frac{1}{k}$ 요소만큼만 달라진다 ($\Rightarrow k = \frac{1}{\varepsilon}$)
- 남은 그래프의 연결된 요소들은 $< k$ 개의 층을 갖는다. DP로 보통 $\approx 2^k \cdot n$ 시간에 풀 수 있음

레크리에이션 알고리즘

- 게임에 대한 많은 알고리즘과 복잡도 [some in SP.268 and our book Games, Puzzles & Computation (2009)]
- $n \times n \times n$ 루빅스 큐브의 직경은 $\Theta \frac{n^2}{\lg n}$ [Demaine, Demaine, Eisenstat, Lubiw, lg n Winslow (2011)]
- 테트리스는 NP-완전 [Breukelaar, Demaine, Hohenberger, Hoogeboom, Kosters, LibenNowell (2004)]
- 풍선을 꼬아 다면체 만들기 [Demaine, Demaine, Hart (2008)]
- 알고리즘적 마법 속임수

MIT의 알고리즘 강의들: (6.006 이후)

- 6.046: 중급 알고리즘 (보다 심화된 알고리즘 & 분석, 적은 코딩)
- 6.047: 계산 생물학 (게놈, 계통발생학 등)
- 6.854: 심화 알고리즘 (전체 분야에 대한 집중적 조사)
- 6.850: 기하학 컴퓨팅 (점, 선, 다각형, 메쉬, . . .)
- 6.849: 기하학적 접기 알고리즘 종이접기, 로봇 팔, 단백질 접힘, . . .
- 6.851: 심화 자료구조 (부대수적 성능)
- 6.852: 분산 알고리즘 (결함이 있는 네트워크에서 일치점 도달)
- 6.853: 알고리즘적 게임 이론 (내쉬균형, 경매 메커니즘 설계, . . .)
- 6.855: 네트워크 최적화 (그래프 최적화: 최단 거리를 넘어서)
- 6.856: 확률적 알고리즘 (난수화가 알고리즘을 어떻게 간단하고 빠르게 만드는가)
- 6.857: 네트워크와 컴퓨터 보안 (암호학)

다른 이론 강의들:

- 6.045: 오토마타, 계산가능성, & 복잡도
- 6.840: 컴퓨팅 이론
- 6.841: 심화 복잡도 이론
- 6.842: 난수성 & 계산
- 6.845: 양자 복잡도 이론
- 6.440: 필수 코딩 이론
- 6.441: 정보 이론

6.006 쿠션의 10 가지 활용법

10. 앉을 수 있음: 상수시간에 영감 받는 것 보장 (기말고사 때 가져올 것)
9. 프리스비 (원형으로 잘라서)*
8. eBay에 한정판으로 팔기 (아마 다시 만들어지지 않을 것—최소 $5)
7. 두 개를 붙여서 상표 가리기* (이 수업을 들었는지 아무도 모를 것임)
6. 휴일의 대화 촉진제… 그리고 침묵 촉진제 (다시 선물하는 것은 추천하지 않음)
5. 점근적으로 최적인 음향 판넬 (피아노 & 기타 연습 다이내믹 프로그래밍)
4. 과녁 맞추기 게임을 위한 연습용 표적
3. 10년 뒤에 여러분이 6.006 에 대해서 기억할 유일한 것(아마 이 10가지 활용법 리스트도)
2. 기말고사 치팅 시트*
1. 세 단어: SNS 프로필 사진