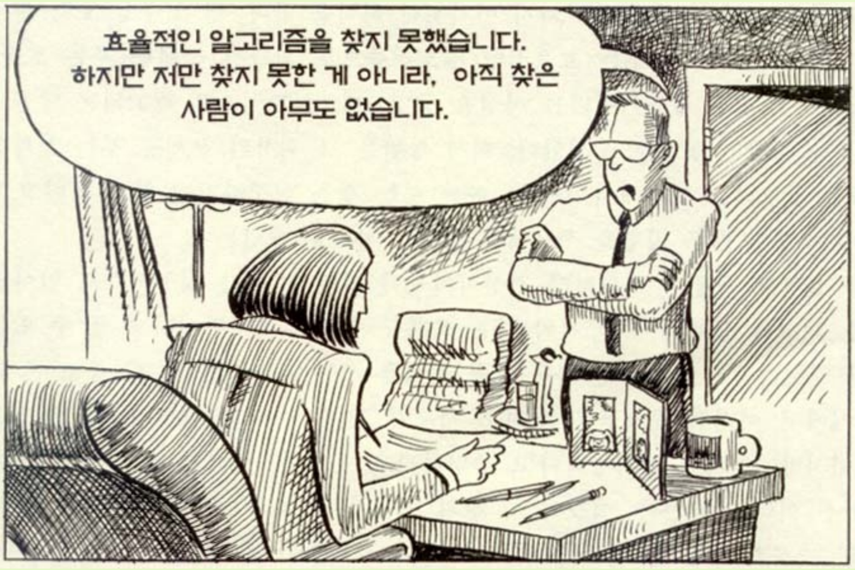
계산복잡도와 다루기 힘든 정도 NP 이론의 소개

알고리즘 해석
강의 슬라이드 #9
도경구



2001 -03 -19 알고리즘 강의 슬라이드 9 2

다차시간 알고리즘 (Polynomial-time Algorithm)

- 입력의 크기가 n일 때, 최악의 경우 수행시간이 $W(n) \in O(p(n))$인 알고리즘.
  - 여기서 $p(n)$은 n의 다차 함수(polynomial function)이다.
- 보기
  - 시간 복잡도가 $2n$, $3n^3 + 4n$, $5n + n^{10}$, $n \lg n$인 알고리즘은 다차시간 알고리즘이다.
  - 시간복잡도가 2^n, $2^{0.01n}$, $2^{\sqrt{n}}$, $n!$인 알고리즘은 다차시간 알고리즘이 아니다.

다루기 힘든 정도(Intractability)

- 다항식으로 시간복잡도가 표시되는 알고리즘을 찾을 수 없는 문제를 "<u>다루기 힘들다(intractable)</u>"라고 한다.
- 반례: 연쇄행렬곱셈문제
  - 무작정알고리즘: $\Theta(2^n)$
  - 동적계획법 알고리즘: $\Theta(n^3)$
  - 따라서 이 문제는 다루기 힘들지 않다 (not intractable)

문제의 일반적인 분류

- 다차시간 알고리즘을 찾은 문제
- 다루기 힘들다고 증명된 문제
- 다루기 힘들다고 증명되지 않았고, 다차시간 알고리즘도 찾지 못한 문제

2001 -03 -19 알고리즘 강의 슬라이드 9 5

1. 다차시간 알고리즘을 찾은 문제

- 모든 알고리즘들이 다차시간 알고리즘
  - ✓ 정렬 문제: $\Theta(n \lg n)$
  - ✓ 정렬된 배열 검색 문제: $\Theta(\lg n)$
  - ✓ 행렬곱셈 문제: $\Theta(n^{2.38})$
- 다차시간이 아닌 알고리즘도 있으나, 다차시간 알고리즘을 찾은 경우
  - ✓ 연쇄행렬곱셈 문제
  - ✓ 최단경로 문제
  - ✓ 최소비용신장트리(Minimum spanning tree) 문제

2001 -03 -19 알고리즘 강의 슬라이드 9 6

2. 다루기 힘들다고 증명된 문제

- 비다항식(nonpolynomial) 크기의 결과를 요구하는 비현실적인 문제
  - ✓ 보기: 모든 해밀토니안 회로를 결정하는 문제
    - 만일 모든 정점들 간에 이음선이 있다면, (n-1)!개의 답을 얻어야 한다.
    - 이러한 문제는 하나의 해밀토니안 회로를 구하는 문제에 비해서 필요이상으로 많은 답을 요구하므로 사실상 비현실적이고, 다루기 힘든 문제로 분류된다.
- 요구가 현실적이나, 다차시간에 풀 수 없다고 증명된 문제
  - ✓ 놀랍게도 이런 부류에 속하는 문제는 상대적으로 별로 없다.
  - ✓ 결정불가능한 문제(Undecidable Problem)
    - 그 문제를 풀 수 있는 알고리즘이 존재할 수 없다고 증명이 된 문제
    - 예: 종료 문제(Halting Problem) 등
  - ✓ 프레스버거 산술(Presburger Arithmetic) 문제
    - Fischer와 Rabin에 의하여 증명됨(1974)

2001 -03 -19 알고리즘 강의 슬라이드 9 7

종료 문제 (Halting Problem)

- 어떤 프로그램 P가 정상적으로 수행되어서 종료하는지를 결정하는 문제
  - ✓ 1936년 Alan Turing에 의해서 증명됨
- **정리**: 종료 문제는 결정불가능하다.
- **증명**: 이 문제를 풀 수 있는 알고리즘이 존재한다고 가정하자. 그 알고리즘은 어떤 프로그램을 입력으로 받아서 그 프로그램이 종료하면 “예”라는 답을 주고, 종료하지 않으면 “아니오”라는 답을 줄 것이다. 그 알고리즘을 Halt라고 하면, 다음과 같은 “말도안돼” 알고리즘을 만들 수 있다.

```
algorithm 말도안돼
   if Halt(말도안돼) == "예" then
      while true do print "야호"
```

(1) 만일 “말도안돼” 알고리즘이 정상적으로 종료하는 알고리즘이라고 한다면, Halt(말도안돼)는 “예”가 되고, 따라서 이 알고리즘은 절대로 종료하지 않는다. 이는 가정과 상반된다.
(2) 만일 “말도안돼” 알고리즘이 정상적으로 종료하지 않는 알고리즘이라고 한다면, Halt(말도안돼)는 “아니오”가 되고, 따라서 이 알고리즘은 종료하게 된다. 이도 마찬가지로 가정과 상반된다.
결론적으로, Halt라는 알고리즘은 존재할 수 없다.

2001 -03 -19 알고리즘 강의 슬라이드 9 8

3. 다루기 힘들다고 증명되지 않았고, 다차시간알고리즘도 찾지 못한 문제

- 많은 문제들이 이 카테고리에 속한다.
  - ✓ 0-1 배낭채우기 문제
  - ✓ 외판원 문제
  - ✓ *m*-색칠하기 문제($m\geq 3$)
  - ✓ 해밀토니안 회로 문제 등등
- 이러한 문제들은 NP(Nondeterministic Polynomial) 문제에 속한다.

2001 -03 -19 알고리즘 강의 슬라이드 9 9

최적화 문제 vs. 결정 문제

- <u>최적화 문제</u>(optimization problem) - 최적의 해를 찾아야 한다.
- <u>결정 문제</u>(decision problem) - 대답이 “예” 또는 “아니오”로 이루어지는 문제 ⇒ 이론을 전개하고 이해하기 쉬움
- 최적화 문제에 대한 해답을 가지고 결정 문제에 대한 해답은 저절로 얻는다.
- 어떤 최적화 문제에 대해서 다차시간 알고리즘이 있다면, 그 알고리즘으로부터 그 문제에 해당하는 결정 문제에 대한 다차시간 알고리즘을 쉽게 유추해 낼 수 있다.
- NP와 P를 다룰 때 결정 문제만 고려해도 충분하다.

2001 -03 -19 알고리즘 강의 슬라이드 9 10

NP 문제의 예

- **외판원 문제**
  - ✓ <u>최적화 문제</u>: 가중치가 있는 방향 그래프에서, 한 정점에서 출발하여 다른 모든 정점을 정확히 한번씩 만 방문하면서 출발점으로 돌아오는 일주여행경로 중에서 총 여행거리가 최소가 되는 경로를 구하시오.
  - ✓ <u>결정 문제</u>: 어떤 양수 *d*가 주어지고, 총 일주여행거리가 *d*를 넘지 않는 경로가 있는지 없는지를 결정하시오.
- **0-1 배낭채우기 문제**
  - ✓ <u>최적화 문제</u>: 배낭에 넣을 아이템의 무게와 가치를 알고 있을 때, 용량이 *W*가 되는 배낭에 아이템을 총 이익이 최대가 되도록 채우시오.
  - ✓ <u>결정 문제</u>: 용량이 *W*가 되는 배낭에 아이템을 총 이익이 최소한 *P*가 되도록 채울 수 있는지를 결정하시오.

2001 -03 -19 알고리즘 강의 슬라이드 9 11

검증(Verification)

- **정의**: P는 다차 시간 알고리즘으로 풀 수 있는 결정 문제들의 집합이다.
  P에 속해 있는 문제들 - 정렬문제, 검색문제, 행렬곱셈문제 등.
  P에 속해 있지 않은 문제 - Presburger Arithmetic
- **검증(Verification):** 결정 문제의 답이 "예"인지를 검증하는 절차. 예를 들어 외판원 결정 문제의 답이 "예"라면, 한 일주여행경로가 주어졌을 때, 그 경로가 과연 그런지를 확인한다.

```
function verify(G: weighted-digraph;
                d: number;
                S: claimed-tour);
  begin
   if S is a tour and the total weight of the edges in S <= d then
      verify := true
   else
      verify := false
  end;
```

이 검증 절차는 다차 시간 안에 수행될 수 있다. 즉, *d*보다 작은 일주여행경로를 찾는 것이 아니라(이 과정은 다차 시간으로 해결하지 못할 수 도 있다), 주어진 일주여행경로가 *d*보다 작게 걸리는 것인지를 알아보는 것이다.

2001 -03 -19 알고리즘 강의 슬라이드 9 12

비결정적알고리즘 (Nondeterministic Algorithm)

1. 추측단계(Guessing state: 비결정적임):
   문제의 입력이 주어지면, 단순히 해답을 추측한다.(말도 안 되는 답이어도 상관없음)
2. 검증단계(Verification state: 결정적임):
   입력: 입력과 추측한 해답
   출력:
   - "맞음"이라는 답을 주고 멈춘다
   - "틀림"이라는 답을 주고 멈춘다
   - 무한 루프

   실제 상황에서 이 비결정적 알고리즘으로 문제를 푸는 것은 불가능하다. 그러나 다음과 같은 경우에는 비결정적 알고리즘이 결정 문제를 "푼다" 고 한다:
   - "예" 답을 줄 입력에 대해서 검증단계가 "맞음" 답을 줄 추측한 해답이 있다.
   - "아니오" 답을 줄 입력에 대해서 검증단계가 "맞음" 답을 줄 추측한 해답이 없다.

2001 -03 -19 알고리즘 강의 슬라이드 9 13

NP의 정의

- **정의:** ***<u>다차시간 비결정적 알고리즘</u>***(Polynomial-time nondeterministic algorithm)이란 <u>검증단계</u>가 다차시간에 수행되는 비결정적 알고리즘을 말한다.
- **정의:** NP(Nondeterministic Polynomial)는 다차 시간 비결정적 알고리즘에 의해서 풀 수 있는 모든 결정 문제의 집합이다.
- 어떤 결정 문제에 대해서 검증을 다차 시간에 하는 알고리즘이 있다면, 그 결정 문제는 NP에 속한다.
- 어떤 결정 문제를 풀 수 있는 다차 시간 알고리즘을 찾을 수 없을 때, 다차 시간 비결정적 알고리즘을 찾으면 그 문제는 NP에 속한다.
- 사색: 그러면 어떤 결정 문제가 주어졌을 때, 그 문제가 NP임을 어떻게 증명할까?

2001 -03 -19 알고리즘 강의 슬라이드 9 14

NP의 특징

- P에 속하는 모든 문제는 당연히 NP에도 속한다.
- NP에 속하지 않는 문제는 어떤 것이 있는가?
  - ✓ 다루기 힘들다(intractable)고 증명이 된 문제, 즉 Halting 문제, Presburger Arithmetic 문제 등이다.
- (중요한 사실) NP에 속한 문제 중에서 P에 속하지 않는다고 증명이 된 문제는 하나도 없다. 따라서 아마도 NP - P = 0일지도 모른다. 그럼 P = NP ? 이것이 사실이라면 거의 모든 결정 문제가 NP에 속하기 때문에 우리가 아는 거의 모든 문제가 다차 시간 알고리즘이 있다는 얘기가 된다. 사실 많은 사람들이 P ≠ NP 일 것 이라고 생각하고 있기는 하지만 아무도 이것을 증명하지 못하고 있는 것이다

2001 -03 -19 알고리즘 강의 슬라이드 9 15

NP-완전(Complete)

- NP-완전 문제에 속하는 문제 중에서 어떤 하나 만이라도 P에 속한다는 것이 밝혀지면, 다른 모든 문제도 P에 속해야 한다.
- CNF(Conjunctive Normal Form): x를 논리변수(logical variable) 라고 하면, x가 참이라는 말은 ξ는 거짓이라는 말과 동일하다. x나 ξ는 리터럴(literal)이라 하고, ∨연산자로 리터럴을 결합하면 절(clause)이라 한다. ∧로 절을 연결하면 CNF가 된다. 예를 들면,
  $(\overline{x_1} \vee x_2 \vee \overline{x_3}) \wedge (x_1 \vee \overline{x_4}) \wedge (\overline{x_2} \vee x_3 \vee x_4)$는 CNF이다.
- CNF-Satisfiability 결정 문제: CNF가 참이 될 수 있도록 논리값(참 또는 거짓)을 지정할 수 있는 지의 여부를 결정하는 문제.
  보기:
  - $(x_1 \vee x_2) \wedge (x_2 \vee \overline{x_3}) \wedge \overline{x_2}$ - yes
  - $(x_1 \vee x_2) \wedge \overline{x_1} \wedge \overline{x_2}$ - no

  이 결정 문제는 NP에 속한다.

2001 -03 -19 알고리즘 강의 슬라이드 9 16

NP-완전 [또]

- **변환(transformation) 알고리즘:** 풀고 싶은 어떤 문제를 A라고 하고, 이미 알고리즘을 알고 있는 어떤 문제를 B라고 하자. A에 대해서 "예"의 답을 해줄 모든 입력을 B에 대해서도 "예"의 답을 해줄 입력으로 변환하는 알고리즘. 그러면, 변환 알고리즘과 B에 대한 알고리즘을 합하면, A에 대한 알고리즘이 나온다.
- **정의:** A에서 B로 다항식시간에 변환하는 알고리즘이 있다면, A는 "P-시간(polynomial-time) 다대일 축소가능(many-one reducible)"하다고 하고, A∝B로 쓴다. 따라서 만일 B문제에 대해서 P-시간 알고리즘이 있고, A에서 B로의 변환 알고리즘도 P-시간이라면, 그 두 알고리즘을 합함으로서 A에 대한 P-시간 알고리즘을 얻게 된다.
- **정리 1:** 결정 문제 B가 P에 속하고 A∝B이면 결정 문제 A는 P에 속한다.

2001 -03 -19 알고리즘 강의 슬라이드 9 17

NP-완전 문제 [또또]

- **정의:** 만일 (1) 문제 B가 NP에 속하고, (2) NP에 속해 있는 모든 다른 문제 A에 대해서 A∝B이면 B는 NP-완전이라고 불리 운다.
- 어떤 문제가 NP-완전인지를 위의 정의에 근거해서 증명하는 일은 매우 어렵다. 왜냐하면 NP에 속한 모든 문제가 그 문제로 축소가능(reducible)하다는 것을 보여야 하기 때문이다. 그러나 다행스럽게도, 1971년 Cook이 다음의 2 정리를 증명했다.
- **정리 2:** (Cook's Theorem) CNF-Satisfiability 문제는 NP-완전이다.
- **정리 3:** 만일 (1) 문제 C가 NP에 속하고, (2) 어떤 NP-완전 문제 B에 대해서 B∝C이면, C는 NP-완전이다.
- Cook에 의해서 CNF-Satisfiability 문제가 NP-완전 임을 알았기 때문에, 주어진 문제가 NP-완전 인지 아닌지는 위의 정리 9.3에 의해서 비교적 쉽게 증명할 수 있다.

2001 -03 -19 알고리즘 강의 슬라이드 9 18