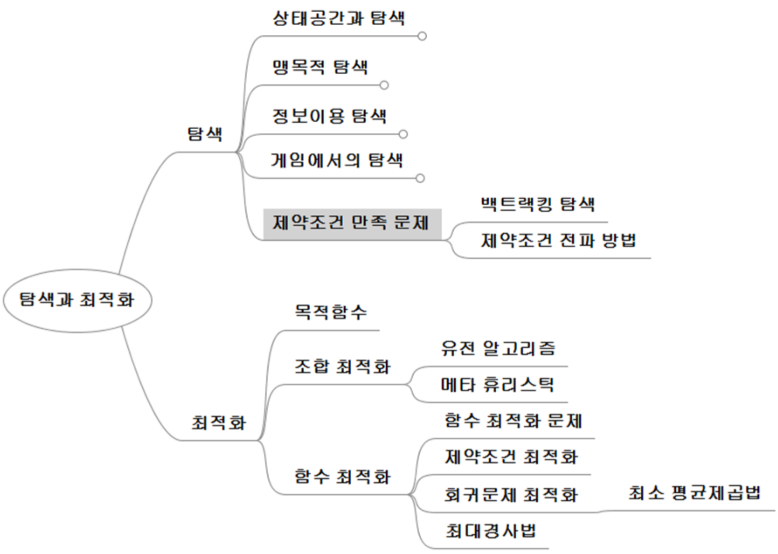
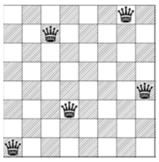
탐색과 최적화 - II

충북대학교 소프트웨어학과
이건명



5. 제약조건 만족 문제

❖ **제약조건 만족 문제**(constraint satisfaction problem)
- 주어진 제약조건을 만족하는 **조합 해**(combinatorial solution)를 찾는 문제
- 예. 8-퀸(queen) 문제

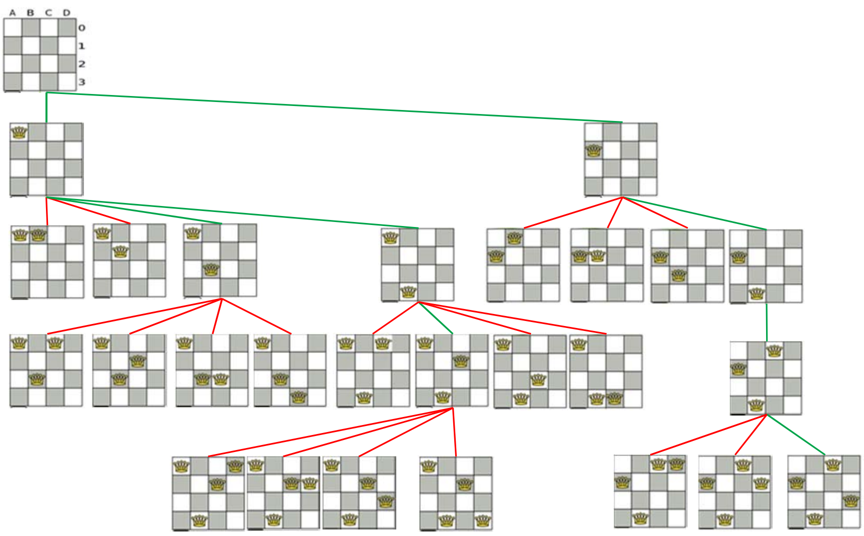


- **탐색 기반의 해결방법**
  - 백트랙킹 탐색
  - 제약조건 전파

- **백트랙킹 탐색(backtracking search)**
  - **깊이 우선 탐색**을 하는 것처럼 **변수**에 허용되는 **값**을 **하나씩 대입**
  - 모든 가능한 값을 대입해서 **만족하는 것이 없으면 이전 단계로 돌아가서** 이전 단계의 변수에 다른 값을 대입

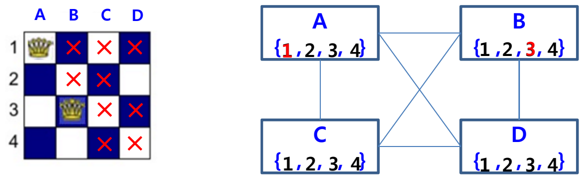
제약조건 만족 문제

❖ 예. **백트랙킹 탐색을 이용한 4-퀸(queen) 문제**



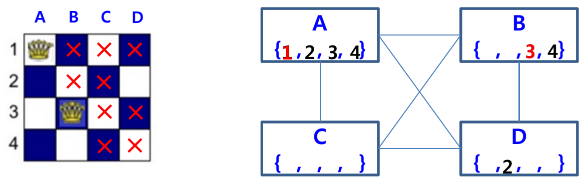
제약조건 만족 문제

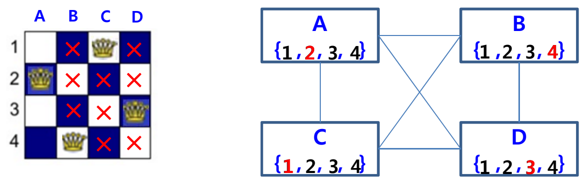
❖ **제약조건 전파(constraint propagation)**
- **인접 변수 간**의 **제약 조건**에 따라 각 변수에 **허용될 수 없는 값**들을 **제거**하는 방식



제약조건 만족 문제

❖ **제약조건 전파(constraint propagation)**
- **인접 변수 간**의 **제약 조건**에 따라 각 변수에 **허용될 수 없는 값**들을 **제거**하는 방식

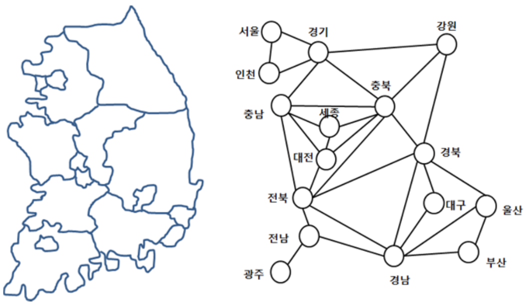




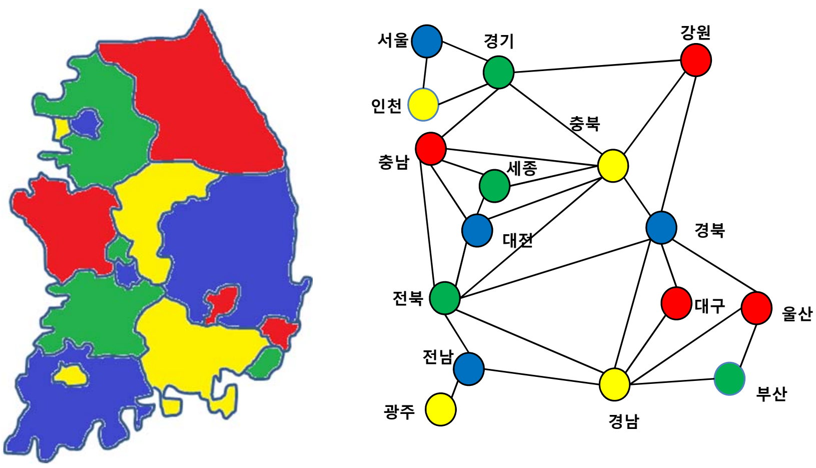
제약조건 만족 문제

❖ **4색 문제(four color problem)**

- 도상에서 인접한 영역 간에는 동일한 색상으로 칠하면 안 되고, 4개 이내의 색상을 사용하여 각 영역을 색칠



제약조건 만족 문제



6. 최적화

❖ **최적화**(optimization)
- 여러 가지 **허용되는 값들 중**에서 주어진 **기준**을 **가장 잘 만족하는 것**을 선택하는 것

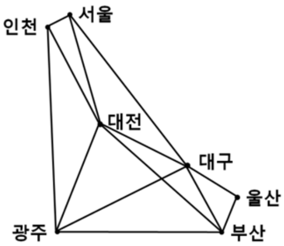
- **목적함수**(objective function)
  - 최소 또는 최대가 되도록 만들려는 함수

- **조합 최적화**
  - 유전 알고리즘
- **함수 최적화**
  - 최대 경사법
  - 제약 함수 최적화

조합 최적화

❖ **조합 최적화(combinatorial optimization)**
- **순회 판매자 문제**(TSP)와 같이 주어진 항목들의 조합으로 해가 표현되는 최적화 문제

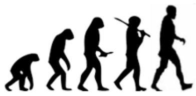
  - 순회 판매자 문제의 목적함수 : 경로의 길이



(서울,인천,광주,부산,울산,대전,서울)

(서울,인천, 대전, 광주,부산,울산, 서울)

유전 알고리즘



❖ **유전 알고리즘**(genetic algorithm, **GA**)
- 생물의 **진화**를 **모방**한 **집단 기반**의 **확률적 탐색** 기법(John Holland, 1975)
- 대표적인 **진화 연산**(evolutionary computation)의 하나
  - 유전 알고리즘, 유전자 프로그래밍(genetic programming), 전화 전략 (evolutionary strategy)
- **생물의 진화**
  - **염색체**(chromosome)의 **유전자**(gene)들이 **개체** 정보 코딩
  - **적자생존**(fittest survival)/**자연선택**(natural selection)
    - **환경**에 **적합도**가 높은 개체의 높은 생존 및 후손 번성 가능성
    - 우수 개체들의 높은 자손 증식 기회
    - 열등 개체들도 작지만 증식 기회
  - **집단**(population)의 진화
    - **세대**(generation) 집단의 변화
  - **형질 유전**과 **변이**
    - 부모 유전자들의 **교차**(crossover) 상속
    - **돌연변이**(mutation)에 의한 변이

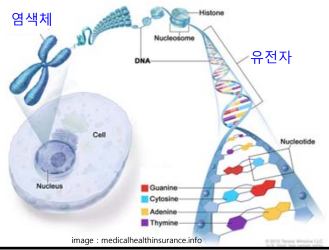
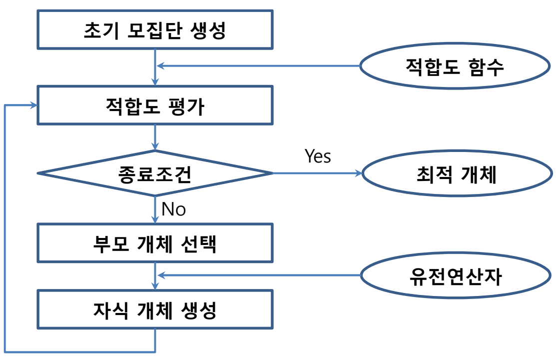


image : medicalhealthinsurance.info

유전 알고리즘

❖ **유전 알고리즘** – cont.
- 생물 **진화**와 **문제 해결**
  - 개체 ⇔ 후보 해(candidate solution)
  - 환경 ⇔ 문제(problem)
  - 적합도 ⇔ 해의 품질(quality)



유전 알고리즘

❖ **유전 알고리즘** – cont.

- **후보해**(candidate solution) **표현**
  - 염색체(chromosome) 표현

| 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 |

(0.6 0.8 . . . 1.2 0.9)

(E2 E5 E3 . . . E11 E7)

- **모집단**(population)
  - 동시에 존재하는 염색체들의 집합

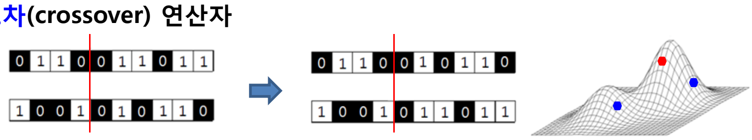
- **적합도 함수**(fitness function)
  - 후보해가 **문제의 해**(solution)로서 **적합한 정도**를 평가하는 함수

유전 알고리즘

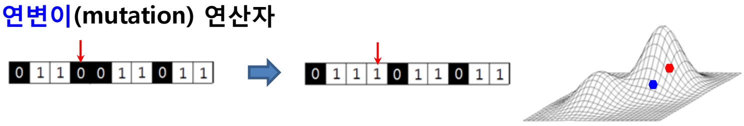
❖ **유전 알고리즘** – cont.

- **부모 개체 선택**(selection)
  - 높은 적합도의 개체가 새로운 개체를 생성할 확률이 높도록 함
  - 적합도에 비례하는 **선택확률**
    - 예. 개체 1의 적합도: 10, 개체 2의 적합도: 5, 개체 3의 적합도: 15

- **유전 연산자**(genetic operator) : 새로운 개체 생성
  - **교차(crossover) 연산자**



  - **돌연변이(mutation) 연산자**



유전 알고리즘

❖ **유전 알고리즘** – cont.
- **세대(generation) 교체**
  - **엘리트주의(elitism)**
    - **우수한 개체를 다음 세대**에 유지

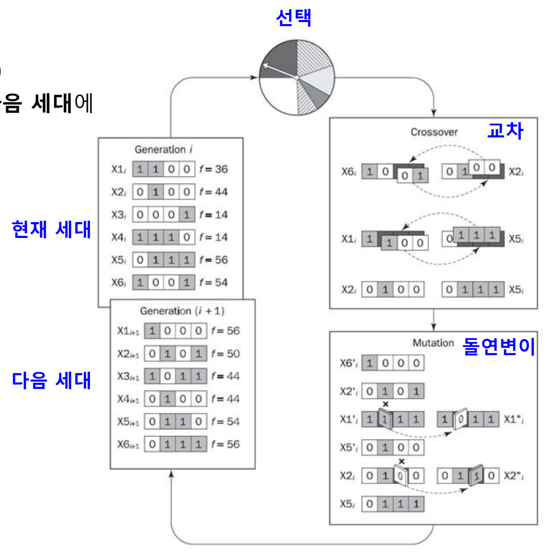


Image : MICHAEL NEGNEVITSKY, Artificial Intelligence

메타 휴리스틱

❖ **메타 휴리스틱**(meta heuristics)
- 최적해(optimal solution)을 보장하지는 않지만 준최적해(suboptimal solution)을 빠르게 찾는 알고리즘

- 유전 알고리즘
- 모방 알고리즘(memetic algorithm)
- 입자 군집 최적화(particle swarm optimization, PSO)
- 개미 집단(ant colony) 알고리즘
- 타부 탐색(Tabu search)
- 담금질 기법(simulated annealing)
- 하모니 탐색(Harmonic search)

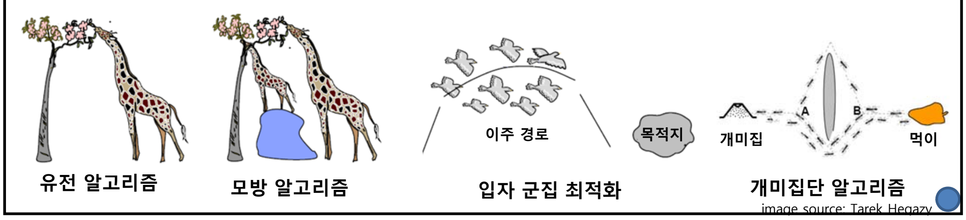
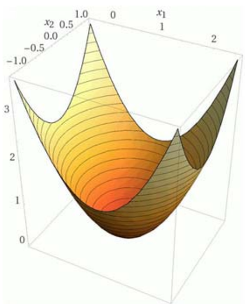


image source: Tarek Hegazy

함수 최적화

❖ **함수 최적화(function optimization)**

- 어떤 **목적 함수**(objective function)가 있을 때, 이 함수를 **최대**로 하거나 **최소**로 하는 **변수 값**를 찾는 최적화 문제



Find x_1, x_2
which minimizes $f(x_1, x_2) = (x_1 - 1)^2 + {x_2}^2$

목적함수 (objective function)

$$\frac{\partial f(x_1, x_2)}{\partial x_1} = 2x_1 - 2 = 0 \qquad x_1 = 1$$

$$\frac{\partial f(x_1, x_2)}{\partial x_2} = 2x_2 = 0 \qquad x_2 = 0$$

$$({x_1}^*, {x_2}^*) = (1,0)$$

함수 최적화

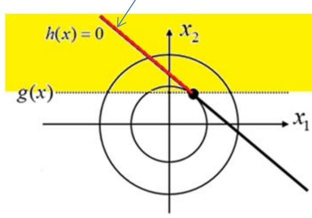
❖ **제약조건 최적화**(constrained optimization)

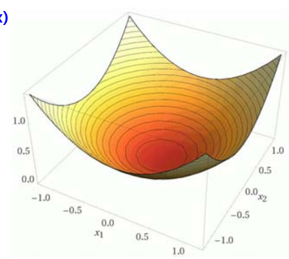
- **제약조건**(constraints)을 만족시키면서 **목적함수**를 **최적화**시키는 변수값들을 찾는 문제

Find x_1, x_2
which minimizes $f(x_1, x_2) = \frac{1}{2}(x_1^2 + x_2^2)$
subject to $h(x_1, x_2) = 1 - x_1 - x_2 = 0$
$g(x_1, x_2) = \frac{3}{4} - x_2 \leq 0$

제약조건
(constraintx)

가능해(feasible solutions)





- 기계학습 방법인 SVM의 학습에서 사용

함수 최적화

❖ **제약조건 최적화**(constrained optimization)

Find x_1, x_2

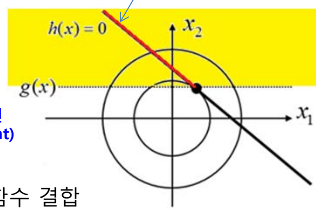
which minimizes $f(x_1,x_2) = \frac{1}{2}(x_1^2 + x_2^2)$

subject to $h(x_1,x_2) = 1 - x_1 - x_2 = 0$

$g(x_1,x_2) = \frac{3}{4} - x_2 \le 0$

가능해(feasible solution)

제약조건 (constraint)



- **라그랑주(Lagrange) 함수** : 제약조건들과 목적함수 결합

$L(x_1,x_2,\lambda,\alpha) = f(x_1,x_2) + \lambda h(x_1,x_2) + \alpha g(x_1,x_2)$ λ, α : 라그랑주 승수(Lagrange multiplier)

$= \frac{1}{2}(x_1^2 + x_2^2) + \lambda(1 - x_1 - x_2) + \alpha\left(\frac{3}{4} - x_2\right)$ $(\alpha \ge 0)$

- **최적화 방법**

$\min_{x_1,x_2 \in FS} f(x_1,x_2) = \min_{x_1,x_2}\max_{\lambda,\alpha} L(x_1,x_2,\lambda,\alpha)$ FS : 가능해(feasible solution)의 집합

$\min_{x_1,x_2}\max_{\lambda,\alpha} L(x_1,x_2,\lambda,\alpha) \ge \max_{\lambda,\alpha}\boxed{\min_{x_1,x_2} L(x_1,x_2,\lambda,\alpha)}$ $L_d(\lambda,\alpha) = \min_{x_1,x_2} L(x_1,x_2,\lambda,\alpha)$

$\ge \max_{\lambda,\alpha} L_d(\lambda,\alpha)$ **쌍대함수(dual function)**

- **쌍대함수를 최대화하면서 상보적 여유성을 만족하는 x_1, x_2를 구함**

$\alpha g(x_1,x_2) = 0$

상보적 여유성(complementary slackness)

함수 최적화

❖ **제약조건 최적화**(constrained optimization) – cont.

$L(x_1,x_2,\lambda,\alpha) = \frac{1}{2}(x_1^2 + x_2^2) + \lambda(1 - x_1 - x_2) + \alpha\left(\frac{3}{4} - x_2\right)$

$L_d(\lambda,\alpha) = \min_{x_1,x_2} L(x_1,x_2,\lambda,\alpha)$

$\frac{\partial L(x_1,x_2,\lambda,\alpha)}{\partial x_1} = x_1 - \lambda = 0 \quad x_1 = \lambda$

$\frac{\partial L(x_1,x_2,\lambda,\alpha)}{\partial x_2} = x_2 - \lambda - \alpha = 0 \quad x_2 = \lambda + \alpha$

$L_d(\lambda,\alpha) = -\lambda^2 - \frac{1}{2}\alpha^2 - \lambda\alpha + \lambda + \frac{3}{4}\alpha$

$\max_{\lambda,\alpha} L_d(\lambda,\alpha)$

$\frac{\partial L_d(\lambda,\alpha)}{\partial \lambda} = -2\lambda - \alpha + 1 = 0 \quad \frac{\partial L_d(\lambda,\alpha)}{\partial \alpha} = -\alpha - \lambda + \frac{3}{4} = 0$

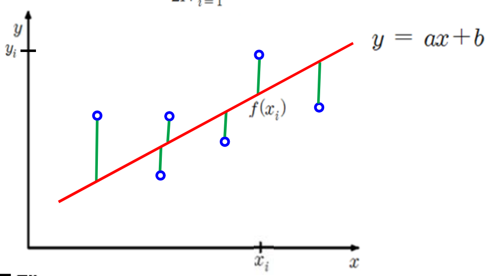
$\lambda = \frac{1}{4} \quad \alpha = \frac{1}{2} \qquad \alpha\left(\frac{3}{4} - x_2\right) = 0 \quad x_2 = \frac{3}{4} \quad 1 - x_1 - x_2 = 0 \quad x_1 = \frac{1}{4}$

$\therefore\ x_1 = \frac{1}{4} \quad x_2 = \frac{3}{4}$

함수 최적화

❖ **회귀(regression) 문제의 최적 함수**
- 주어진 **데이터**를 가장 잘 **근사**(近似, approximation)하는 **함수**
- **최소 평균제곱법**(least mean square method)
  - **오차 함수**(error function) 또는 **에너지 함수**(energy function)를 최소로 하는 함수를 찾는 방법

$$E = \frac{1}{2N}\sum_{i=1}^{N}(y_i - f(x_i))^2$$



  - **최적화 문제**

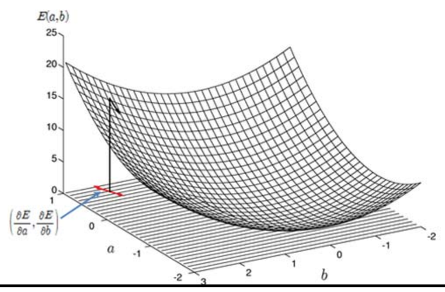
$$\text{Find } a,b \text{ which minimizes } \min_{a,b} \frac{1}{2N}\sum_{i=1}^{N}(y_i - ax_i - b)^2$$

함수 최적화

❖ **최대 경사법**(gradient descent method, **경사 하강법**)
- 함수의 **최소값 위치**를 찾는 문제에서
  오차 함수의 **그레디언트**(gradient) 반대 방향으로 조금씩 움직여 가며
  최적의 파라미터를 찾으려는 방법
- **그레디언트**
  - 각 파라미터에 대해 편미분한 벡터 $\left(\frac{\partial E}{\partial a}, \frac{\partial E}{\partial b}\right)$
- 데이터의 입력과 출력을 이용하여 각 파라미터에 대한 그레디언트를 계산하여 **파라미터**를 **반복적**으로 **조금씩 조정**

$$a^{(t+1)} = a^{(t)} - \eta \frac{\partial E}{\partial a}$$

$a^{(t)}$: 현 시점에서 파라미터 a의 값
η : 학습율 (0 < η < 1)



함수 최적화

- ❖ **최대 경사법**(gradient descent method)
  - ▪ 회귀모델, 신경망 등의 기본 학습 방법
  - ▪ 국소해(local minima)에 빠질 위험
  - ▪ 개선된 형태의 방법 존재
    - • conjugate gradient method 등