KPC 정보처리기술사 핵심 오브젝트

[Module 10]

Algorithm 이론과 전략

ITPE Information Technology Professional Engineer

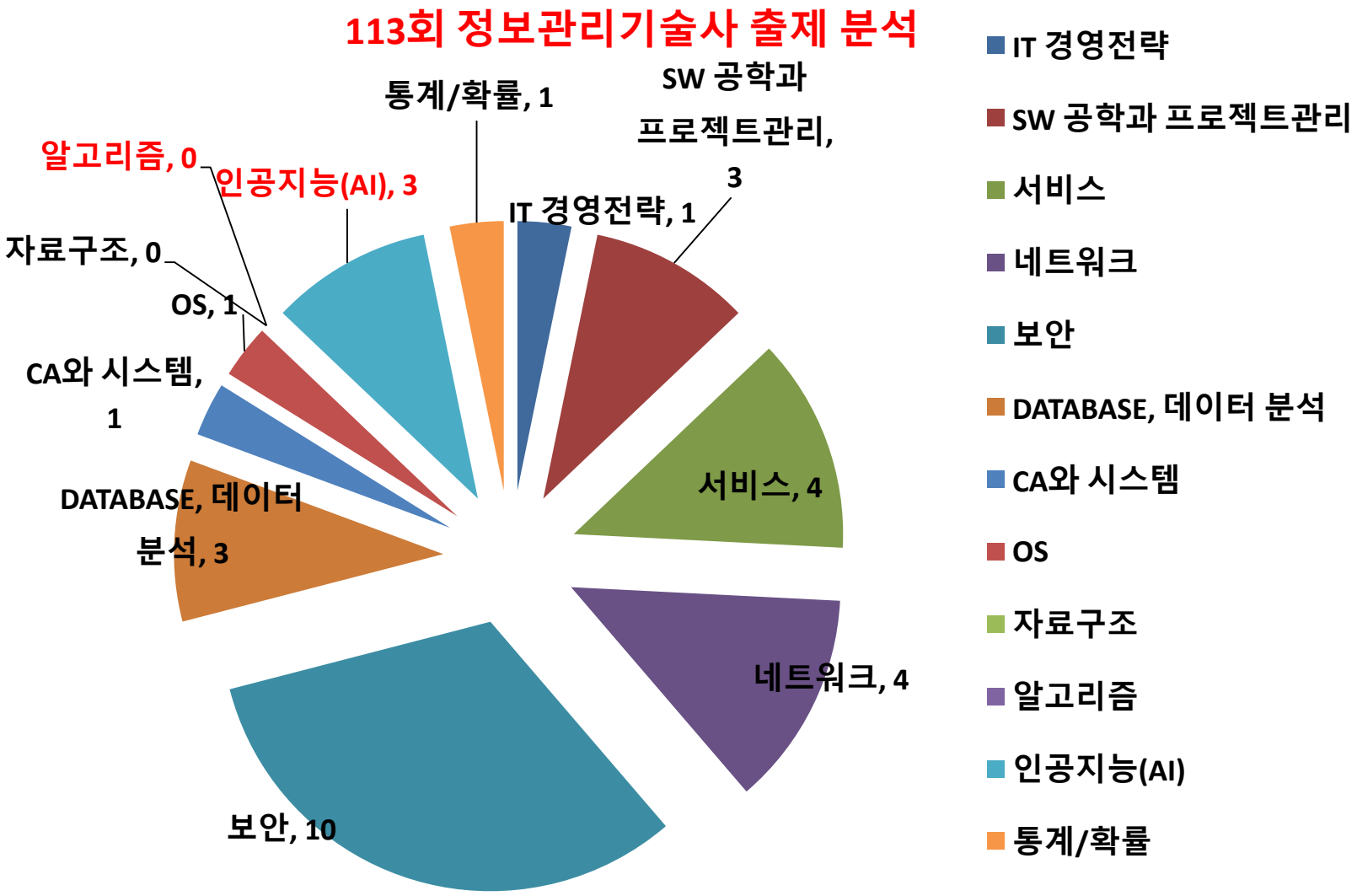
kpc

목차

분류	출제빈도	Page	분류	출제빈도	Page
Chapter I. 알고리즘			Chaper II. 압축기술		
1. 알고리즘	중		압축기술	중	
2. O-Notation	중		동영상 압축 표준	중	
3. 재귀 알고리즘	중		MPEG	중	
4. 알고리즘 평가 방법	중		HEVC	중	
정렬(Sorting)	상				
5. Bubble Sort	상				
6. Quick Sort	상				
7. 그래프(Graph)	중	42			
8. 최소 신장 트리(MST)	하				
Prim 알고리즘	하				
Kruskal 알고리즘	하				
9. 최단 경로 탐색 (Dijkstra 알고리즘)	상				
10. Hash	상				

Algorithm & AI 출제 경향

I. 113회 정보관리기술사 기출문제



I. 113회 정보관리기술사 기출문제

회차	교시	도메인	문제
113	1	인공지능	MCTS(Monte Carlo Tree Search)
113	2	인공지능	인공 신경망의 가장 단순한 형태인 퍼셉트론(Perceptron)의 구조와 활성화 함수(activation function)에 대하여 설명하고, 단층 퍼셉트론으로는 Exclusive-OR 연산을 학습할 수 없는 이유를 설명하시오.
113	3	인공지능	유전 알고리즘(Genetic algorithm)을 정의하고, 그 알고리즘의 흐름을 설명한 후, 이 알고리즘에서 사용되는 기법인 룰렛 휠 선택(roulett wheel selection)의 역할에 대하여 설명하시오.

I. 111회 정보관리기술사 기출문제

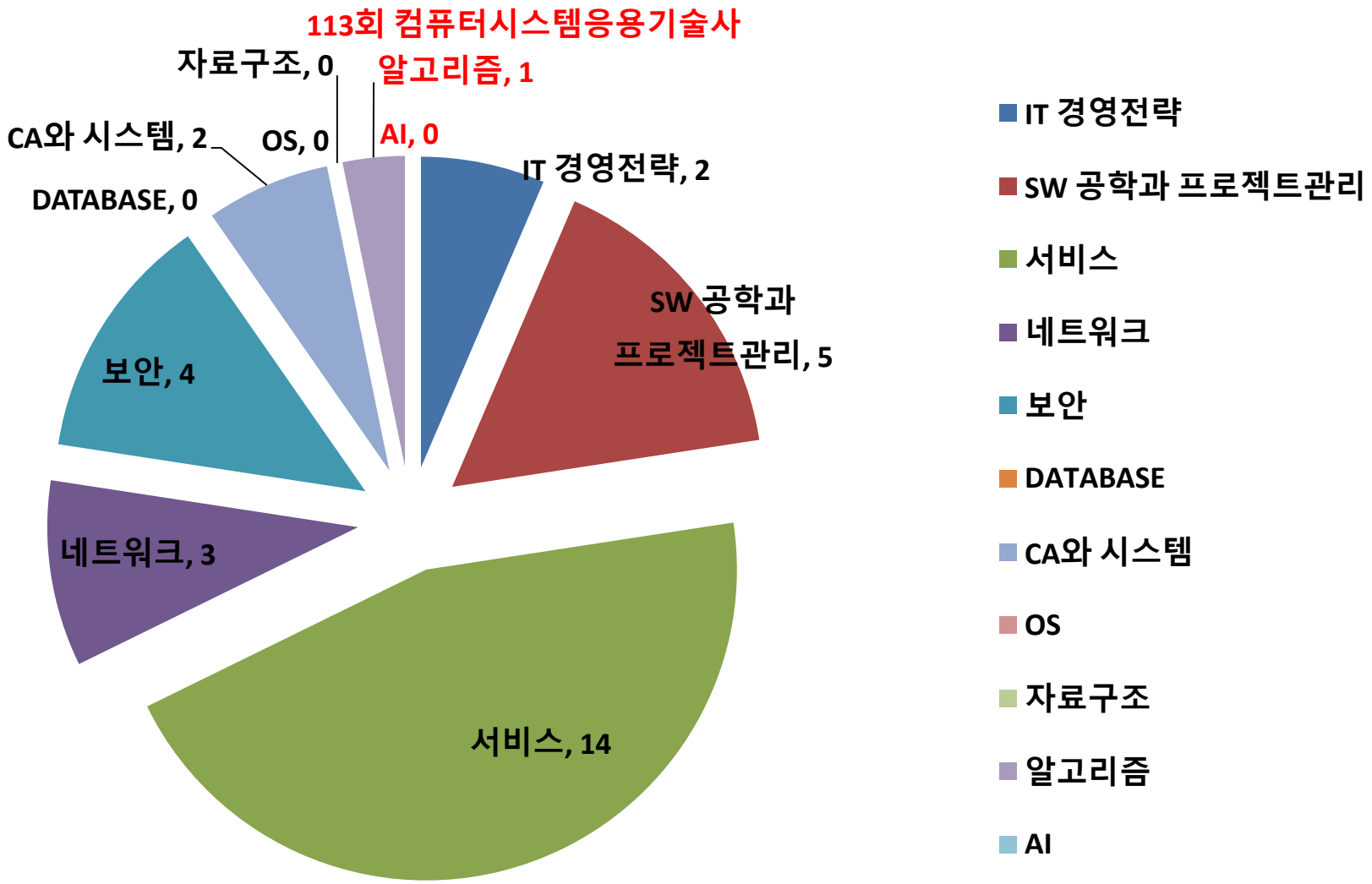
회차	교시	도메인	문제
111	2	인공지능	2. 딥러닝(Deep Learning) 구현을 위한 Feed Forward 모델을 설명하고, CNN(Convolutional Neural Network)과의 차이점을 설명하시오.
111	4	인공지능	1. 데이터 분석의 분류 모델에서 모델 평가 시 사용되는 혼동행렬(Confusion Matrix)에 대하여 설명하고, 혼동행렬 기반의 모델 성능평가 항목을 산식을 이용하여 설명하시오.
111	4	인공지능	4. 최근 기업들은 인공지능(AI) 콜센터를 구축하려고 한다. 인공지능 콜센터는 딥러닝(Deep Learning)을 적용한 인공지능 상담사가 고객과 대화하고 요청된 업무를 처리하는 시스템이다. 도입 시 적용 가능한 서비스를 인바운드, 아웃바운드, 녹취록 분석으로 구분하여 구현 방안을 제시하시오.

[Feed Forward 모델]
가장 간단한 방법의 인공신경망 방법으로, 신경망 정보가 입력 노드에서 은닉노드를 거쳐 출력 노드까지 전달 되며 순환 경로가 존재하지 않는 그래프를 형성.

[혼동행렬(Confusion Matrix)]란
클래스 분류의 결과를 정리한 표이다. 주로 알고리즘의 성능을 평가할 때 평가하는 지표로 많이 사용.
Precision-Recall Curve(프리스즌-리콜, 정확도와 검출율)
- 검출율: Recall 이라는 용어 사용/클래스: 유사한 특징을 갖는 데이터들이 속하는 집합 혹은 군

Algorithm & AI 출제 경향

2. 113회 컴퓨터시스템응용기술사



2. 113회 컴퓨터시스템응용기술사

회차	교시	도메인	문제
113	4	인공지능	양자 알고리즘이 현대 암호에 미치는 영향에 대해서 'Shor 알고리즘'과 'Grover 알고리즘' 중심으로 설명하시오.

2. 111회 컴퓨터시스템응용기술사

회차	교시	도메인	문제
111	1	인공지능	6. CNN(Convolution Neural Networks)

I. 110회 정보관리기술사 기출문제

회차	교시	도메인	문제
110	1	AI	12. 챗봇(chatbot)에 대해 설명하시오.
110	3	AI	2. 인공지능 산업 발전을 위한 (1) 촉진 관련 법제도, (2) 규제 관련 법제도를 기술하고, 현행 법제도의 문제점과 개선방안에 대하여 설명하시오.

2. 110회 컴퓨터시스템응용기술사

회차	교시	도메인	문제
110	4	AI	10. 딥 러닝(Deep Learning : 심층학습)에 대해 설명하시오.

ITPE Information Technology Professional Engineer

최근 출제 문제

기출문제

108회 기술사

1. 그리디(Greedy) 알고리즘에 대하여 다음 질문에 답하시오. 230원어치를 구매함, 손님이 이때
   가. 지폐 1000원을 받고 동전으로 770원을 돌려 줄 때 최소 동전수를 찾는 그리디 알고리즘을 설명하시오. (단, 동전의 액면은 500원, 100원, 50원, 10원임)
   나. 위 알고리즘을 C 또는 Java 언어로 구현하시오.
   (정보관리, 108회 3교시 1번)

2. 인공지능의 특이점(Singularity)에 대하여 설명하고, 엑소브레인(Exobrain) 소프트웨어와 딥뷰(Deepview)의 개념과 기술 요소에 대하여 설명하시오.
   (정보관리, 108회 3교시 6번)

3. 기계 학습(Machine Learning)의 개념 및 응용 분야.
   (컴시응, 108회 1교시 1번)

4. 딥러닝(Deep Learning) 기술을 정의하고, 최근 기업에서 딥러닝을 응용하는 예를 2가지 제시하시오.
   (정보관리, 107회 2교시 2번)

최근 출제 문제

기출문제

105회 정보관리기술사

1. Quick Sort 알고리즘에 대하여 설명하고 아래의 C언어 소스 코드에서 필요시 함수등을 추가하여 완성하시오(단, Sort 순서는 오름차순)

```
#include <stdio.h>
#include<string.h>

void QuickSort(int *data, int n)
{
}
void main()
{
 char data[8] = {'B','I','D','O','Z','L','H'};
 puts(data);
 QuickSort(data,7);
 puts(data);
}
```

(정보관리, 105회 4교시 6번)

최근 출제 문제

기출문제

104회 컴퓨터시스템응용기술사

1. Bubble sort 알고리즘에 대하여 다음 내용을 설명하시오.(오름차순 기준)

   1) flag를 두지 않는 경우와 flag를 두는 경우로 나누어 설명하고, flag를 두는 이유를 설명하시오.

   2) 다음 키 값을 갖는 파일을 Bubble sort 알고리즘을 적용하여 sort하는 과정을 보이시오.

      (n=8 : 30, 50, 10, 80, 40, 60, 70, 90)

기존 기출 문제

영역	구분	회차	교시	문 제
그래프	조직응용	72회	3교시	6. Graph의 모든 Vertex를 방문하는 두가지 방법을 설명하고 이중에 최단 거리를 찾는 경우 어느 방법이 적합한지 설명하라.
그래프	정보관리	84회	3교시	5. 다음 그래프에 Dijkstra가 제안한 최소비용(least-cost)경로설정 알고리즘이 적용되는 과정을 제시하시오.(그래프의 에지옆 숫자는 비용을 의미함)
그래프	정보관리	87회	4교시	2. 다음과 같이 7개 신도시의 도로 공사를 최소의 비용으로 설계할 때, 다음 물음에 답하시오. 단, 노드는 도시 이름을 나타내고 간선은 공사 비용이다. (1) MST(Minimal Spanning Tree)의 개념에 대해 설명하시오. (2) 위 그래프의 비용 인접 리스트를 도직화하시오. (3) Prim 알고리즘을 이용해서 MST를 구하는 절차를 보이시오. 단, 시작 노드는 A이다. (4) Kruskal 알고리즘을 이용해서 MST를 구하는 절차를 보이시오.
그래프	정보관리	90회	4교시	3. 임의로 배열된 8개 숫자를 1에서 8까지 순서대로 배열하는 8-퍼즐 게임이 있다. A* 알고리즘을 8-퍼즐 게임에 적용하여 설명하시오.
그래프	정보관리	96회	1교시	1. 그래프를 이용하여 최적 경로를 찾는 데 이용되는 최소 신장 트리 (Minimal spanning tree) 알고리즘에 대하여 설명하시오.
그래프	정보관리	98회	4교시	1. 인공지능 응용 정보시스템이나 게임 개발 시 길 찾기에 이용되는 Djkstra 알고리즘과 A* 알고리즘을 비교 설명하시오.
알고리즘 성능 분석	정보관리	89회	4교시	1. 알고리즘의 평가 방법인 Time Complexity와 Space Complexity에 대해 설명하시오.
분할 정복	정보관리	75회	1교시	6. 알고리즘 설계기법에서의 분할통치법(Divide and conquer)을 설명하시오
동적 계획법	정보관리	87회	1교시	13. 알고리즘 설계 기법 중 동적 계획법(Dynamic Programming)에 대해 설명하시오.
탐욕 알고리즘	정보관리	83회	4교시	3. 컴퓨터 알고리즘 중 Greedy method에 관해서 물음에 답하시오. 1) Greedy method의 특징과 해를 구하는 프로세스 절차에 관해서 설명하시오. 2) knapsack problem에서 n=3(objects수), m=20(knapsack capacity),그리고 (p1,p2,p3)=(25,24,15), (w1,w2,w3)=(18,15,10)일 때 물음에 답하시오. 단, p1,p2,p3은 profit이고, w1,w2,w3은 weight

Chapter I. 알고리즘

정의

알고리즘은 다음의 조건을 만족해야 한다.

- 입력 : 외부에서 제공되는 자료가 0개 이상 존재한다.
- 출력 : 적어도 1개 이상의 결과를 내어야 한다.
- 명확성 : 수행 과정은 명확하고 모호하지 않은 명령어로 구성되어야 한다.
- 유한성(종결성) : 유한 번의 명령어를 수행 후(유한 시간 내)에 종료한다.
- 효율성 : 모든 과정은 명백하게 실행 가능(검증 가능)한 것이어야 한다.

- 복잡도.
  - O(1) : 입력 자료의 수에 관계없이 일정한 실행 시간을 갖는 알고리즘
  - O(log N) : 입력 자료의 크기가 N일경우 $\log_2 N$ 번만큼의 수행시간을 가진다.
  - O(N) : 입력 자료의 크기가 N일경우 N 번만큼의 수행시간을 가진다.
  - O(N log N) : 입력 자료의 크기가 N일경우 $N*(\log_2 N)$ 번만큼의 수행시간을 가진다.
  - $O(N^2)$: 입력 자료의 크기가 N일경우 N^2 번만큼의 수행시간을 가진다.
  - $O(N^3)$: 입력 자료의 크기가 N일경우 N^3 번만큼의 수행시간을 가진다.
  - $O(2^n)$: 입력 자료의 크기가 N일경우 2^N 번만큼의 수행시간을 가진다.
  - O(n!) : 입력 자료의 크기가 N일경우 n*(n-1)*(n-2)... * 1(n!) 번만큼의 수행시간을 가진다.

1. Algorithm
2. O-Notation
3. 재귀 알고리즘
4. 알고리즘 평가방법
5. Bubble Sort
6. Quick Sort
7. Dijkstra(다익스트라) 알고리즘

1. 알고리즘(Algorithm)

Algorithm

❑ "Factorial n"을 구하는 재귀(Recursive) 알고리즘을 작성하시오.

(단, 프로그래밍 언어에는 제한이 없음, Factorial n(n!): n * (n-1) * (n-2) * ... * 1)

(관리 90회 1교시)

I. 알고리즘(Algorithm)의 정의 및 학습목적

가. 알고리즘(Algorithm)의정의

- 어떤 작업을 수행하기 위해 입력을 받아 원하는 출력을 만들어 내는 과정
- 사전적 의미는"어떤 문제를 해결해 나가는 특별한 방법"또는"문제 해결을 위해 컴퓨터로 이용될 수 있는 명확한 방법"
- 문제 해결 절차를 체계적으로 기술한 것
- 특정한 작업을 수행하기 위한 유한명령어들로 구성되며 컴퓨터의 수행에 적합한 문제 해결을 위한 방법

나. 알고리즘(Algorithm)의 학습 목적

- 특정한 문제를 위한 알고리즘의 습득
- 체계적으로 생각하는 훈련
- 지적 추상화의 레벨 상승

Intellectual abstraction: 연구나 개발에 있어 정신적 여유를 유지하기 위해 매우 중요한 요소

II. 알고리즘(Algorithm)의 조건

조건	설명
입출력	외부에서 0개 이상 입력을 받아 1개 이상 출력을 생성한다
명확성	각 단계가 단순해야 하며, 모호하지 않고 이해하기 쉬움 특정 프로그래밍 언어로 변환 하는데 어려움을 느끼지 않을 정도
유한성	한정된 수의 작업 후에는 반드시 끝나야 함.
효과성	모든 명령이 수행 가능해야 함
효율성	궁극적으로 컴퓨터를 통해서 구현되는데 충분히 큰 입력에 대하여 알고리즘의 수행시간은 크게 차이가 발생 가능하기 때문에 효율적으로 수행되어야 함

- 입출력, 명확성, 유한성, 효과성을 만족하는 알고리즘이 존재하면 그 문제가 해결 가능하지만 궁극적으로 컴퓨터에서 수행하는 것을 목표로 하므로 실용성이 있어야 하고 효율성도 필요

-단순히 원하는 결과를 얻을 수 있는 알고리즘이 아닌, 처리시간이나 기억장소 사용측면에서 효율성 있는 알고리즘을 개발하는 것이 주요 목표.

ITPE Information Technology Professional Engineer

2. O-notation(O-표기법)

O-notation

❑ "Factorial n"을 구하는 재귀(Recursive) 알고리즘을 작성하시오.
(단, 프로그래밍 언어에는 제한이 없음, Factorial n(n!): n * (n-1) * (n-2) * ... * 1)
(관리 90회 1교시)

I. O-notation (O-표기법)

가. 알고리즘의 복잡도

- 점근적인 복잡도 평가: 알고리즘에서 문제의 크기를 나타냄, 처리할 자료 항목의 수로서 수행시간에 많은 영향을 끼침.
- 다항식의 차수이거나 탐색이나 정렬될 파일의 크기, 그래프의 노드 수

나. O-Notation (O-표기법, Big O)

- 데이터 수 n에 대해서 복잡도가 어떠한 함수로 나타나는가를 간단히 표현하기위한 알고리즘
- 너무 자세하게 분석하는 것도 문제가 되기 때문에 적절한 단순화가 필요
- 알고리즘의 소요시간이 입력의 크기n에 대해 O(n^2)라면 기껏해야 n^2에 비례하는 시간이 소요됨

복잡도표기	설명
O(1)	상수형, 입력 크기와 무관하게 바로 해를 구함. 해쉬
$O(\log_N)$	로그형, 입력 자료를 나누어 그 중 하나만 처리
O(N)	선형, 입력 자료를 차례로 하나씩 모두 처리
$O(N\log_N)$	분할과 합병형, 자료를 분할하여 각각 처리하고 합병
$O(N^2)$	제곱형, 주요처리(기본연산) loop 구조가 2중인 경우
$O(N^3)$	세제곱형, 주요처리(기본연산) loop 구조가 3중 경우
$O(2^n)$	지수형, 가능한 해결방법 모두를 다 검사하며 처리함

[알고리즘계산시간:O(1) < O(logN) < O(n) < O(NlogN) < O(N^2) < O(N^3)

알고리즘 수행시간 분석은 아래과 같은 3가지 표기법을 보통 쓴다.

- **Big O 표기법**
- **Big Omega 표기법**
- **Big Theta 표기법**

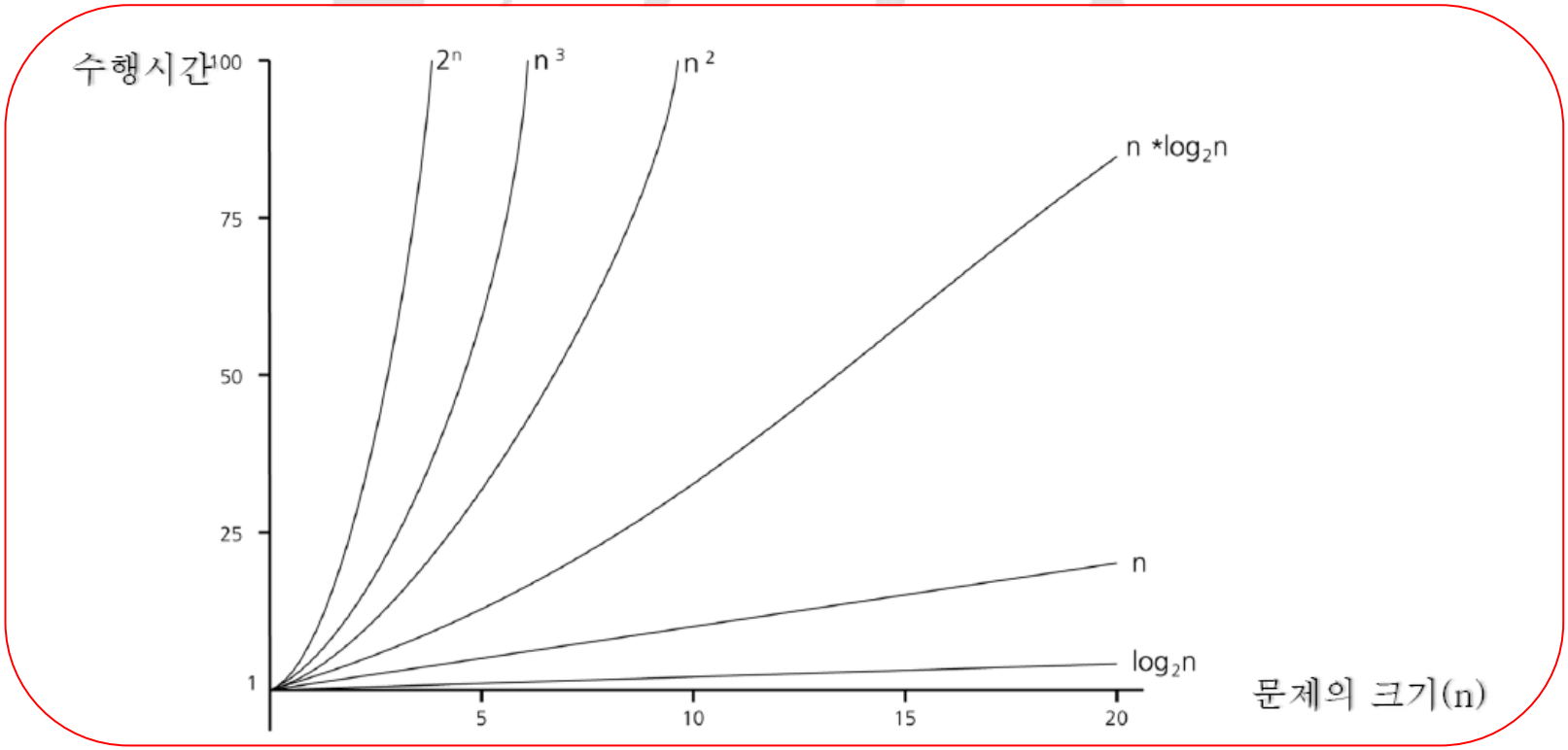
ITPE Information Technology Professional Engineer

- O(f(n))의 점근적 증가율이 f(n)을 넘지 않는 모든 함수들의 집합

$$O(g(n)) = \{f(n) \mid \exists c > 0, n_0 > 0 \; s.t \; \forall n \geq n_0, \; f(n) \leq cg(n)\}$$

풀어서 말하면

$$O(g(n)) = \{f(n) \mid \text{모든} n \geq n_0 \text{에 대하여}, \; f(n) \leq cg(n) \text{의 양의 상수} c \text{와} n_0 \text{가 존재한다}\}$$



ITPE Information Technology Professional Engineer

3. 재귀 알고리즘

재귀 알고리즘

❑ "Factorial n"을 구하는 재귀(Recursive) 알고리즘을 작성하시오.

(단, 프로그래밍 언어에는 제한이 없음, Factorial n(n!): n * (n-1) * (n-2) * ... * 1)

(관리 90회 1교시)

I. 자기 자신으로부터 호출되는 함수, 재귀함수 – 이를 적용한 알고리즘을 재귀 알고리즘의 개념

가. Recursive Algorithm(재귀 알고리즘) 의 정의

- 수학적 귀납법을 역순으로 이용하여 자신의 함수를 반복적으로 호출하는 방법

나. 재귀알고리즘의 구성

분할 정복	문제의 크기가 N인 큰 문제를 작은 문제로 환원, 이때 작은 문제 역시 큰 문제와 유사함
재귀 호출	Self Call, Boomerang
아주 작은 문제	직접 해결할 정도로 작아짐, 베이스 케이스가 됨

II. 재귀 알고리즘의 작성 절차 및 효율성

가. 재귀 알고리즘의 작성 절차

Step1	- 더 작은 문제로 표시할 수 있는지 시도(문제 크기를 하나씩 줄이는 방법, ,반으로 줄이는 방법, 다른 여러 개의 작은 문제의 조합으로 표시하는 방법) - 문제 크기 파라미터 N을 확인
Step 2	문제를 직접 풀 수 있는 것이 어떤 경우인지 베이스 케이스 확인
Step 3	- N이 줄어서 반드시 베이스 케이스를 만나는지 확인 - N이 양수인지 음수인지, 짝수인지 홀수인지, 또는 부동소수인지 정수인지 모든 경우에 대해 모두 검증.
Step 4	베이스 케이스와 베이스 케이스가 아닌 경우를 나누어서 코드를 작성

나. 재귀 알고리즘의 효율성

- 공간적 비효율(저장공간)
- 시간적 비효율(저장, 복원에 걸리는 시간)
- 가능하다면 반복문으로 대치하는 것이 유리

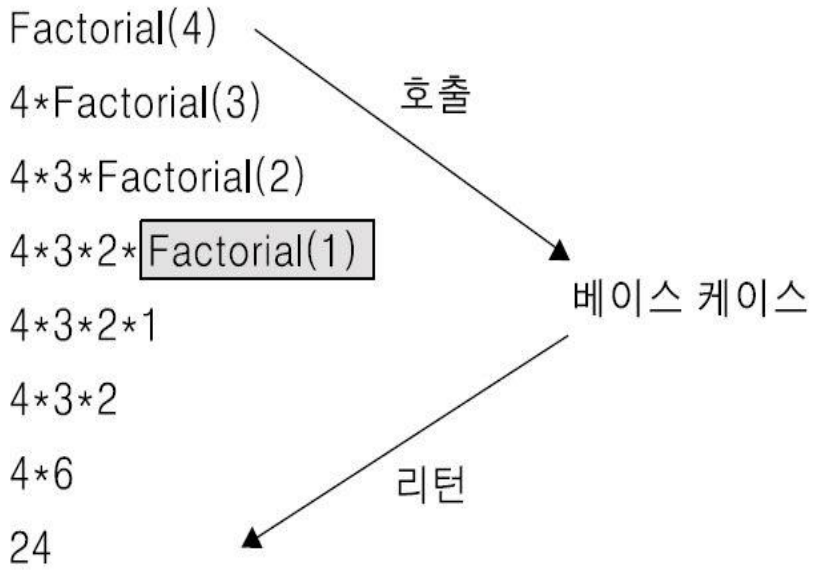
ITPE Information Technology Professional Engineer

III. 문제에서 제시한 Factorial n 에 대한 풀이

가. 문제에 대한 해석

n! = n • (n-1) • (n-2) • ... • 1 (단, 1! = 0! = 1)

```
int Factorial(int n)
{ if (n = = 1)              /* 종료 조건 */
    return 1;
  else
    return(n * Factorial(n-1));  /* 재귀 호출 */
}
```



나. 코드로 구현

```
int Factorial (int n)                    팩토리얼
{ int product = 1;                       곱셈의 결과 값을 초기화
  for (int i = 1; i <= n; i++)           1부터 n까지
    product *= i;                        계속 곱해서 저장
  return product;                        결과를 리턴
}
```


ITPE Information Technology Professional Engineer

IV. 재귀 알고리즘의 이용

가. 이진 탐색 : 문제 크기의 감소, N, N/2, N/4, ... , 1

나. 문자열 뒤집기

다. K 번째 작은 수 찾기

라. 피보나치 수열

ITPE Information Technology Professional Engineer

4. 알고리즘 평가방법

알고리즘 평가방법

❑ 알고리즘의 평가 방법인 Time Complexity와 Space Complexity에 대해 설명하시오.
(관리 89회 4교시)

I. 문제를 해결하기 위한 방법, 알고리즘의 개요

가. 알고리즘(Algorithm)의 정의

- 주어진 문제를 해결하기 위한 잘 정의된 동작들의 집합

나. 알고리즘의 유형

- 알고리즘 분석의 결과는 입력된 자료의 수 N에 대하여 수행 시간을 함수 관계로 표현한 형태

다. O 표기법

- 입력 자료의 수 N에 대하여 그 실행 시간 함수 T(N)은 일정한 유형을 나타나며, 상수의 요소를 없앤 입력 자료 수(N)을 따르는 실행 시간의 추이만을 나타낸 방법

II. 알고리즘의 소요량과 알고리즘 유형

가. 소요량

구분	특징 및 설명
Time complexity	- 어떤 알고리즘이 얼마만큼의 시간을 필요로 하는가 측정 - 디스크 가격 및 메모리 가격이 저가로 되면서, 현재는 공간 소요량보다 시간 소요량이 중요함(시간으로 측정)
Space complexity	- 어떤 알고리즘이 얼마만큼의 공간(메모리 혹은 디스크)를 필요로 하는가 측정 - 보통 시간소요량이 적은 알고리즘은 공간 소요량이 많고 공간소요량이 적은 알고리즘은 시간 소요량이 큼

[표-1]소요량(복잡도) 유형

ITPE Information Technology Professional Engineer

나. 알고리즘의 유형

유형	특징 및 설명
1	- 입력 자료의 수와 상관없이 일정한 수행시간을 갖는 알고리즘 - 하나의 프로그램에서 자주 사용하지 않는 경우에 해당됨
logN	- 입력 자료의 수(N)에 따라 실행 시간이 logN의 관계를 만족하는 경우에 해당 - N이 증가함에 따라 실행시간이 조금씩 증가함. - 매우 성능이 좋은 알고리즘에 해당됨
N	- 선형적인 입력 자료의 수에 따라 선형적으로 N배가 되는 구조 - 2배로 입력하면 2배로 수행시간을 갖는 구조
NlogN	- N < NlogN <N^2 알고리즘 형태로 - N이 두배면 수행기간이 2배가 조금 넘어감
N^2	- 보통 for Loop에 해당하는 구조로 - 입력 자료의 수가 N이면 제곱에 해당하는 수행시간을 갖음
N^3	- 삼중 루프에 해당
2^N	- 입력 자료 수 N이 수행되면 기하급수적으로 증가하는 수행 시간

ITPE Information Technology Professional Engineer

Ⅲ. 시간 소요량과 공간소요량에 대한 사례

가. 시간 소요량(N^2에 해당하는 사례)

```
for i:= 1 to n
    for j;= 1 to n
        A[i,j] = A[i,j]* 2
```

- 위의 간단한 코드는 각각의 A[I,j]를 2배 해주는 사례
- n*n 행렬 A는 n^2 의 요소 -> 각각의 요소 (n^2개)의 계산을 필요로 time complexity 는 n^2

나. 공간 소요량

```
for i:= 1 to n
    for j;= 1 to n
        A[i,j] = A[i,j]* 2
```

- 위의 간단한 코드는 각각의 A[I,j]를 2배 해주는 사례
- N^2의 행렬 구조로서, 저장공간은 N^2의 공간이 필요함

ITPE Information Technology Professional Engineer

5. Sort

Bubble Sort

1. 퀵 정렬(Quick sort)알고리즘을 설명하고, 다음 데이터를 퀵 정렬 알고리즘을 사용해서 정렬하는 과정을 설명하시오. 30,15,16,24,38,33,17,29,32 (정보관리 99회 4교시)

2. 다음은 c언어로 작성된 버블정렬(Bubble sort)알고리즘 프로그램을 완성하시오.

```
#include <stdio.h>
Int main() {
  Int data[5]={2,5,1,4,3};
  Bubble(data,5);

For (int i=0; i<5; i++) {
  Print("%d", data[i]);
    }
 Return 0;
}
```

(관리 92회 2교시)

5. Sort

Bubble Sort

3. Bubble sort 알고리즘에 대하여 다음 내용을 설명하시오.(오름차순 기준)

1) flag를 두지 않는 경우와 flag를 두는 경우로 나누어 설명하고, flag를 두는 이유를 설명하시오.

2) 다음 키 값을 갖는 파일을 Bubble sort 알고리즘을 적용하여 sort하는 과정을 보이시오.

(n=8 : 30, 50, 10, 80, 40, 60, 70, 90)

(컴시응, 104회 2교시)

I. 정렬 알고리즘의 구분

가. 정렬 알고리즘의 구분

- 정렬 알고리즘을 선택할 때는 사용되는 컴퓨터 시스템의 특징, 정렬할 자료의 양, 초기 자료의 배열 상태, 키 값의 분포 상태, 작업 공간의 크기, 자료의 이동 횟수, 키의 비교 횟수 등을 고려함.
- 내부 정렬(Internal Sorting): 데이터의 크기가 주 기억장소 용량보다 적을 경우 기억장소를 활용하여 정렬
- 외부 정렬(External Sorting): 데이터의 크기가 주 기억장소 용량보다 클 경우 외부 기억장치를 사용 정렬

키워드	내용 정리/요약
정의	주어진 문자열들을 임의의 조건에 따라 순서대로 앞에서부터 순차적으로 나열된 구조 또는 방식
구분	- 내부 정렬(Internal Sorting): 데이터의 크기가 주 기억장소 용량보다 적을 경우 기억장소를 활용하여 정렬 - 외부 정렬(External Sorting): 데이터의 크기가 주 기억장소 용량보다 클 경우 외부 기억장치를 사용 정렬
내부정렬	삽입, 버블, 퀵, 쉘, 선택, 힙(Heap), 병합(Merge) 정렬, 기수(Radix) 정렬
외부정렬	균형, 다단계, 교대, 계단식

ITPE Information Technology Professional Engineer

정렬 알고리즘의 상세구분

구분	정렬 알고리즘의 상세설명	
내부 정렬 (Internal Sorting)	버블 정렬 (Bubble)	- 인접한 두 개의 원소를 비교하여 자리를 교환하는 방식으로 정렬 알고리즘 중에서 가장 효율성이 떨어지는 알고리즘 이지만 구현이 간단한 정렬 알고리즘.
	선택 정렬	- 카드 뭉치를 정렬 시에 가장 작은 카드를 찾아서 제일 앞에 놓고, 그 다음 작은 카드를 찾아서 다음에 놓고, 반복 정리하는 정렬 방법(Selection Sort)
	삽입 정렬	- 자료 배열에서, 모든 요소를 앞에서부터 차례대로 이미 정렬된 배열 부분과 비교하여, 자신의 위치를 찾아 삽입하는 정렬 방법(Insert)
	쉘	- 주어진 리스트를 적당한 매개변수 값만큼 떨어진 레코드와 비교하여 교환하는 과정을 반복하는 방법
	퀵	- 분할 및 정복법에 기반 전체 리스트를 2개로 분할 후 각각 다시 정렬
	Heap	- 레코드를 트리구조인 힙형태로 변환, 가장 큰 레코드 키 값을 가지는 루트 노드를 출력한 후 나머지 트리를 다시 힙으로 구성하는 방법
	2-way merge	- 주어진 레코드들을 각각 크기가 1인 정렬된 리스트로 간주, 순서에 맞추어 반복적으로 합병함으로써 최종적으로 1개의 리스트를 생성
	기수	- 숫자의 각 자리수를 기준으로 정렬

ITPE Information Technology Professional Engineer

정렬 알고리즘의 상세구분

구분	정렬 알고리즘의 상세설명	
외부 정렬 (External Sorting)	균형합병정렬	- 정렬하고자 하는 입력 파일을 몇 개의 테이프로 균등하게 분배하여 합병하는 방법 - 각 테이프에는 블록들이 들어가 있고, 이 블록들을 각각 주기억장치에서 정렬하여, 다시 여러 개의 테이프에 분산 저장하는 방법
	다단계 합병	- 입력 파일의 분배 시에 피보나치 수열을 이용하여 분배하는 정렬방법
	교대합병	- 자기테이프의 순방향과 역방향 모두 가능한 상태에서 사용
	계단식 합병	- 복사량을 줄이기 위한 불균형 방법 중 하나로 연속 병합 정렬이라고 함 - 부분적으로 정렬된 서브파일들을 한 개의 빈 테이프에 병합해 정렬하는 방법을 반복하여 정렬하는 방법입

ITPE Information Technology Professional Engineer

I. 구현이 간단한 정렬 알고리즘 버블 정렬

가. 버블정렬or 거품 정렬(Bubble sort)의 정의

인접한 두 개의 원소를 비교하여 자리를 교환하는 방식으로 정렬 알고리즘 중에서 가장 효율성이 떨어지는 알고리즘 이지만 구현이 간단한 정렬 알고리즘.

나. 버블정렬의 방법

Step1	2 , 5, 1, 4, 3 2, 5, 1, 4, 3 2, 1, 5, 4, 3, 2, 1, 4, 5, 3 2, 1, 4, 3, 5
Step2	2, 1, 4, 3, 5 1, 2, 4, 3, 5 1, 2, 4, 3, 5 1, 2, 3, 4, 5 1, 2, 3, 4, 5
: Step5	… 원소의 수만큼 Step 반복.

►첫 번째 원소부터 마지막 원소까지 인접한 두 원소를 비교하여 정렬이 필요할 경우 두 값을 swap.

► 마지막 원소 까지 비교 후 처음부터 다시 인접한 원소 비교를 반복.

► 원소의 개수만큼 반복하여 정렬 완성.

ITPE Information Technology Professional Engineer

II. 버블 정렬 알고리즘 구현

```
Void bubble(int a[ ], int n) {
  int i, j, temp;

  for (i=0; i<n-1; i++) {
      for (j=1; j<n-1; j++) {
          if (a[j-1] > a[j]) {
          temp = a[j-1];
          a[j-1] = a[j];
          a[j] = temp;
          }
       }
   }
}
```

버블 정렬 알고리즘 이해

N은 정렬 원소의 개수

i = 0, i 가 n-1이 되면 i 루프 종료

j = 1, j 가 n-1이 되면 j 루프 종료.

a[j-1] > a[j] 이면 스왑. 아니면 다음 동작 진행.

[버블 정렬 개선]

int s 추가

s에 0과(정렬되어 있지 않는 경우0, s가 1(정렬되어 있으면 break;) 몇 가지 추가 문장 처리

I. 선택정렬(Selection Sort)

- 가장 간단한 정렬 알고리즘: 일상 생활에서 가장 많이 사용하는 방법으로, 카드 뭉치를 정렬 시에 가장 작은 카드를 찾아서 제일 앞에 놓고, 그 다음 작은 카드를 찾아서 다음에 놓고... 카드 정리
- 많은 비교, 적은 교환

데이터	3 2 5 1 4 6
1st Round	배열중 가장 작은 값을 찾아 기준위치인 첫번째 배열과 자리교환 3 2 5 1 4 6 -> 1 2 5 3 4 6 1이 3의 위치로, 3이 1의 위치로 교환
2nd Round	정렬된 (첫번째) 값을 빼고 가장 작은 값을 찾아 기준위치인 두번째 값과 자리교환 1 2 5 3 4 6 -> 1 2 5 3 4 6 1을 제외한 가장작은 수 2는 기준위치이므로 자기자신의 위치 그대로 둠
3rd Round	정렬된(첫번째, 두번째) 값을 빼고 나머지중 가장 작은 값을 찾아 기준위치인 세번째 값과 자리교환 1 2 5 3 4 6 -> 1 2 3 5 4 6 1, 2를 제외한 나머지중 가장작은수 3을 기준위치의 5와 자리교환
4th Round	정렬된(첫번째, 두번째, 세번째) 값을 빼고 나머지중 가장 작은 값을 찾아 기준위치인 네번째 값과 자리교환 1 2 3 5 4 6 -> 1 2 3 4 5 6 1, 2, 3을 제외한 나머지중 가장작은수 4을 기준위치의 5와 자리교환
5th Round	정렬된(첫번째, 두번째, 세번째, 네번째) 값을 빼고 나머지중 가장 작은 값을 찾아 기준위치인 다섯번째 값과 자리교 1 2 3 4 5 6 -> 1 2 3 4 5 6 1, 2, 3, 4를 제외한 나머지중 가장작은수 5는 기준위치이므로 자기자신 위치 그대로 둠
정렬결과	1 2 3 4 5 6

가. 가장 작을 값을 찾아

나. 변수 min에 저장

(n-1까지; 전체)

다. 교환

[소스]

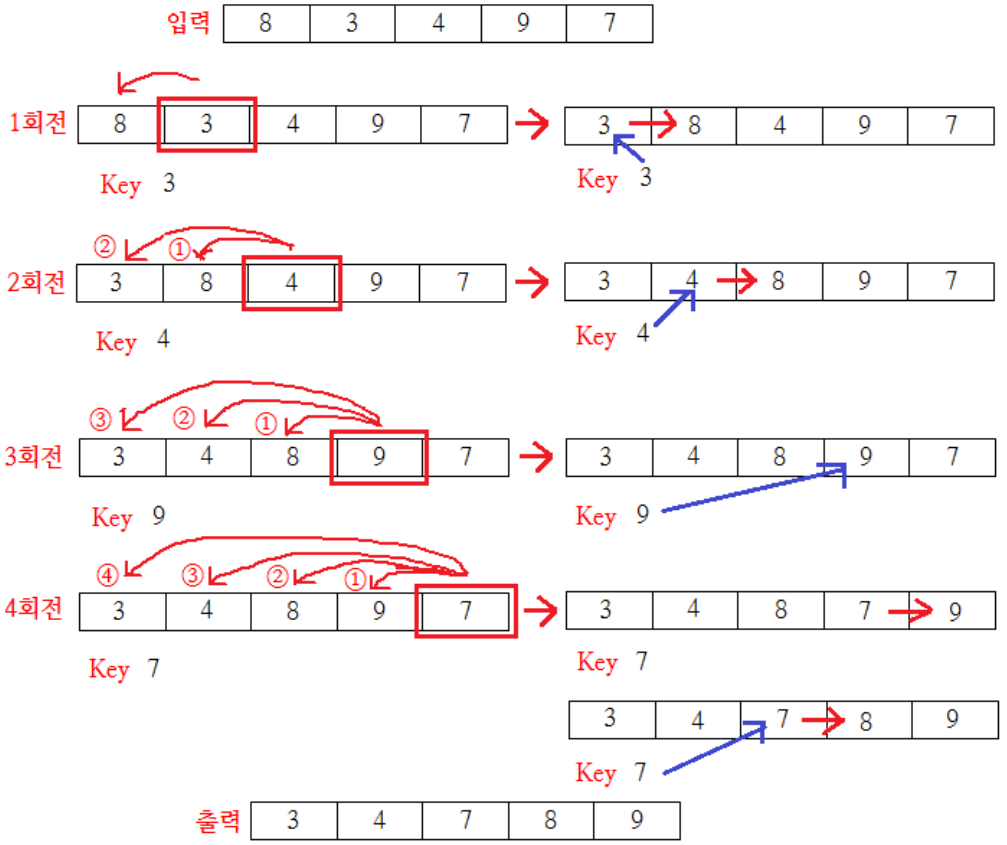
```
Void select_sort(int a[], int
        n)

For(i=0; i<n-1;i++)
{
  For(j=i+1;j<n; j++)

  ....
}
```

I. 삽입 정렬(Insertion Sort)

- 자료 배열에서, 모든 요소를 앞에서부터 차례대로 이미 정렬된 배열 부분과 비교하여, 자신의 위치를 찾아 삽입하는 정렬 방법
- 이미 정렬된 배열이나, 대충 정렬된 배열에 대해서는 뛰어난 성능을 보이지만, 그렇지 않은 경우에는 속도가 매우 느림.
- 인접한 요소만 교환하는 삽입 정렬의 단점.

[설명]

1. 배열 int a[ ] 와 크기 N을 가질 때,
   1) i = 1;
   2) j = i;
   3) a[j-1] > a[i]이고, j > 0인 동안
      (예) 8 > 3
      a[j] = a[j-1]
      (예) a[1] = 8
      j를 하나 감소한다.
   4. a[j] = a[i] // 빈 공간에 a[i]를 삽입한다.
      (예) a[0] = 3
   5. i를 하나 증가하고, 2로 돌아간다.



VI. 쉘 정렬(Shell Sort)

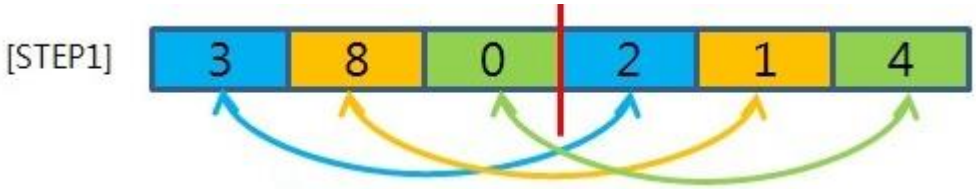
삽입 정렬은 한칸씩 이동하면서 비교하기 때문에 만일 가장 멀리 떨어진 곳에서 비교가 이루어진다면 이동 시 많은 오버헤드가 발생할 수 있다.

그 단점을 보완하기 위해 만들어진 정렬로서, $O(N^{1.25})$으로 측정

* 정렬한 배열에서 h 간격으로 요소들을 뽑아서, 삽입 정렬

3	8	0	2	1	4

1. 주어진 배열에 대해 반으로 쪼갠 후, 쪼갠 블록을 각 쌍으로 삽입 정렬을 시행한다. -> h가 10으로...
   (h가 10 이면, 0, 10, 20 , 30 .... 요소들을 뽑아서 이 요소들간의 삽입 정렬을 한다.)



[설명] 주어진 배열에 대해 반으로 쪼갠 후, 쪼갠 블록을 각 쌍으로 삽입 정렬을 시행한다.
예를 들면 {3,2}, {8,1}, {0,4} 가 쌍으로 삽입 정렬이 실행되고 결과는 {2,3}, {1,8}, {0,4} 로 나타날 것이다.

ITPE Information Technology Professional Engineer

2. 즉, h 가 1, 11, 21..... -> 삽입 정렬

3. 위 정렬이 한번 끝나면, h를 적당히 줄여, 이번엔 5로 정하면,
   0, 5, 10, 15 요소들을 뽑아서, 삽입정렬.

4. H가 1이 될까지 정렬하면, 아마~ 거의 다 정렬이 되어 있는 상태겠죠.

5. 이 상태에서 매우 빠른 속도로 삽입 정렬할 수 있음.

   이유: 삽입 정렬된 배열이나 대충 정리된 배열에서는 매우 뛰어난 성능을 보이므로, 이 방법을 사용

ITPE Information Technology Professional Engineer

6. Quick sort

출제예상문제

1. 퀵 정렬(Quick sort)알고리즘을 설명하고, 다음 데이터를 퀵 정렬 알고리즘을 사용해서 정렬하는 과정을 설명하시오.
   30,15,16,24,38,33,17,29,32
   (정보관리, 99회 4교시)

2. Quick Sort 알고리즘에 대하여 설명하고 아래의 C언어 소스 코드에서 필요시 함수 등을 추가하여 완성하시오(단, Sort 순서는 오름차순)

```
 #include <stdio.h>
 #include<string.h>
void QuickSort(int *data, int n)
{
}
void main()
{
 char data[8] = {'B','I','D','O','Z','L','H'};
 puts(data);
 QuickSort(data,7);
 puts(data);
}
```

Void QuickSort(int data[ ], int n)

(정보관리 105회, 4교시 6번)

ITPE Information Technology Professional Engineer

I. 분할과 정복의 대표적인 퀵 정렬(Quick Sort) 알고리즘의 설명

가. 퀵 정렬(Quick Sort) 알고리즘의 개념

- 입력자료를 특정한 키(제어키)보다 작은 값을 갖는 자료들과 큰 값을 갖는 자료로 분리하여 한 개의 파일을 논리적으로 두 개의 파일로 재배열 하고 각각 파일에 대해 순환적으로 퀵 정렬을 사용
- 정렬하기 위해 리스트 분리 기준이 되는 임의의 한 원소(Pivot)을 선정
- Pivot 원소를 기준으로 입력 리스트를 2개의 부분 리스트로 분할
- 정렬 과정에서 각 부분 리스트에서 다시 퀵 정렬을 수행

나. 퀵 정렬(Quick Sort)의 특징

- 평균 수행능력이 좋은 정렬 방법, 평균 수행시간은 O(nlogn)
- 수행과정에서 순환 알고리즘을 사용하며 스택 공간이 필요하며 분할과 정복기법을 순환적으로 사용
- 제자리 정렬 알고리즘 이지만 키들의 상대적 위치가 유지되지 않는 불안정한 정렬 알고리즘
- Pivot 원소의 선정 방법에 따라 정렬의 수행시간이 달라지며, 두 부분으로 나누었을 때 양쪽 리스트의 크기가 같다면 최상의 상태가 됨.

ITPE Information Technology Professional Engineer

[알고리즘 설명] 오름차순 정렬

1) 우측 M 먼저 Pivot으로 선택하고, 퀵 정렬을 시작한다.

T	O	L	E	A	R	N	S	O	R	T	A	L	G	O	R	I	T	H	M

2. 좌측에서부터는 M보다 큰 값이 있나를 i를 증가하면서 검사하고, 구간의 우측부터는 M보다 작은 값이 있나를 j를 감소시키면서 검사한다. (while 문장으로, i 하고, j 수행)

	i																	j	
T	O	L	E	A	R	N	S	O	R	T	A	L	G	O	R	I	T	H	M

3. T와 H를 비교해서, H가 작으므로, 두 값을 바꾼다.

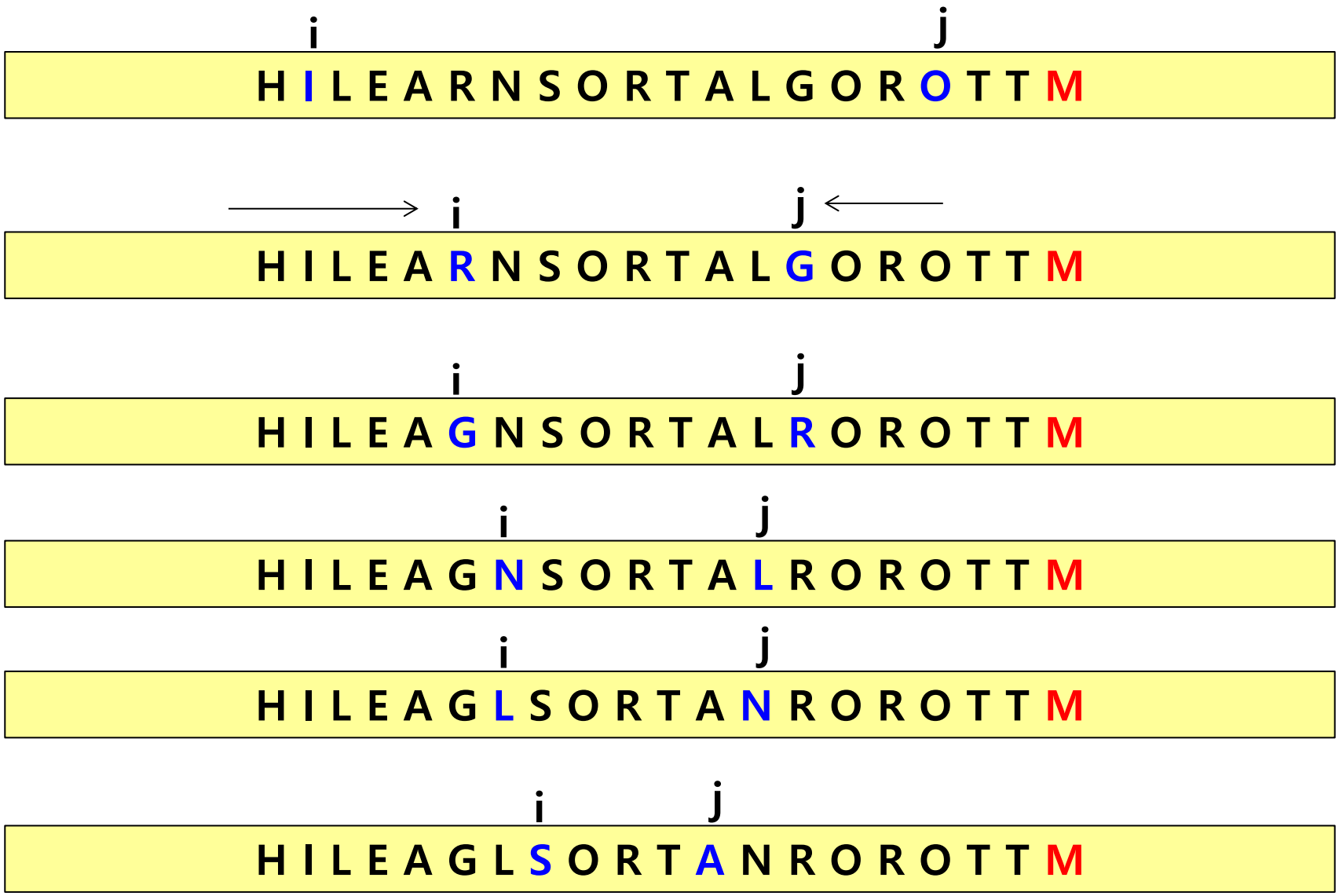
	i																	j	
H	O	L	E	A	R	N	S	O	R	T	A	L	G	O	R	I	T	T	M

4. 다음으로 걸린 값은 O 와 I 이다. 이것들도 바꾼다.

① →	i																	j	
H	O	L	E	A	R	N	S	O	R	T	A	L	G	O	R	I	T	T	M

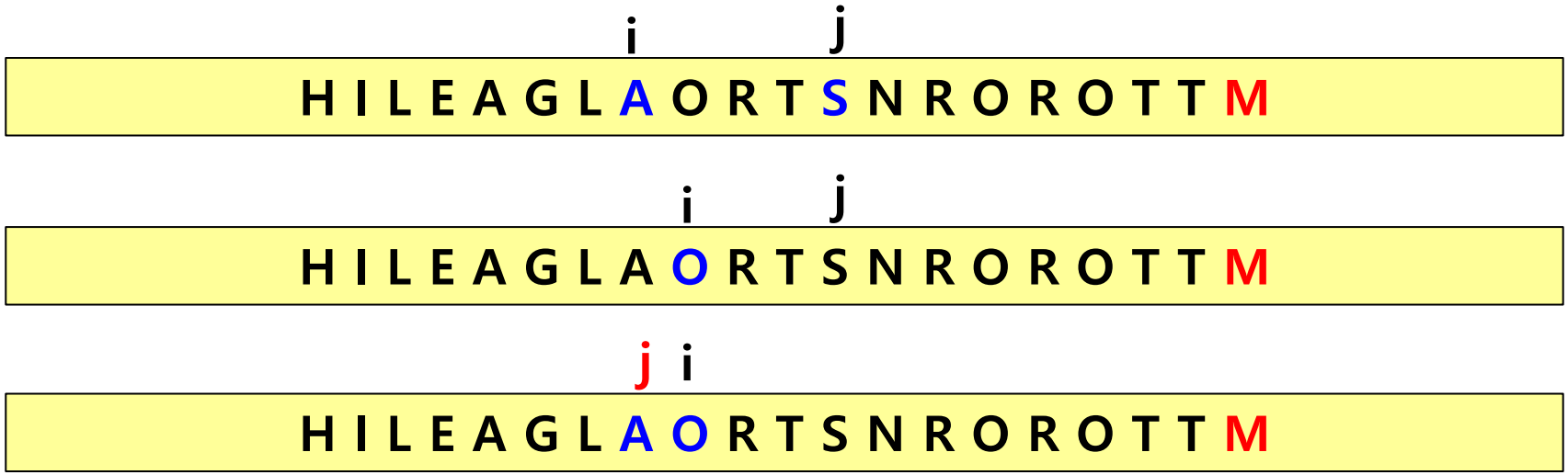
ITPE Information Technology Professional Engineer

[알고리즘 설명]

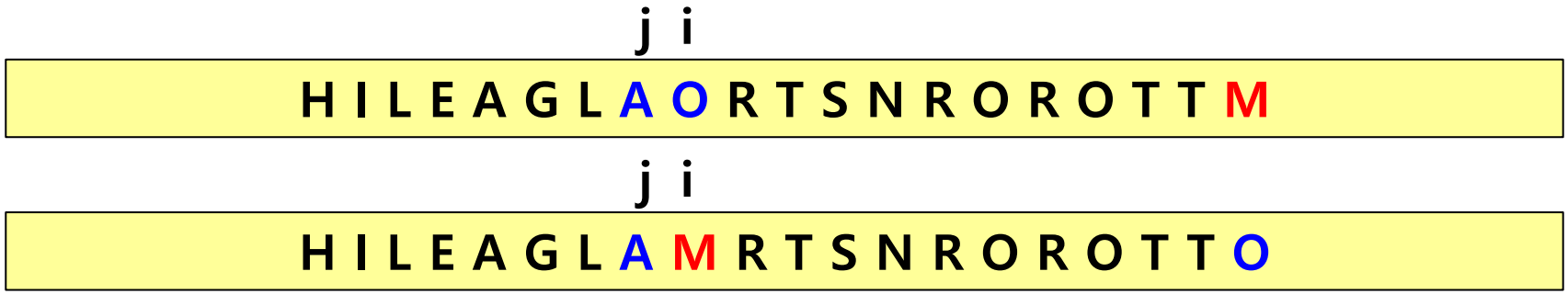


ITPE Information Technology Professional Engineer

[알고리즘 설명]



5. i는 O를 가리키게 되고, j는 A의 위치를 가리키게 된다. 이렇게 i와 j의 위치가 같거나, i가 j보다 우측으로 가게 될 때에는 분할이 완료된 것이므로, i의 자리에 있는 O와 M을 마지막으로 교환한다.



이제 배열은 M을 축으로 해서, 좌측은 모두 M 보다 작은 값이고, 우측은 모두 M보다 큰 값임을 알 수 있다.

[알고리즘 설명]

6. 좌측의 H와 A까지의 구간을 A를 축값으로 해서 분할하며, 우측의 R와 O까지 구간은 O을 축값으로 해서 분할한다.

   오를쪽 소구간 R의 시작 주소는 위 5번 절차에서, a+i+1 이다.

7. 이러한 과정을 크기가 1이 될 때까지 반복하고, 정렬은 완료하게 된다. 참고로 구간의 구간을 나누고 하는 과정이 재귀적으로 수행됨을 알 수 있다.

ITPE Information Technology Professional Engineer

[C로 구현한 퀵정렬]

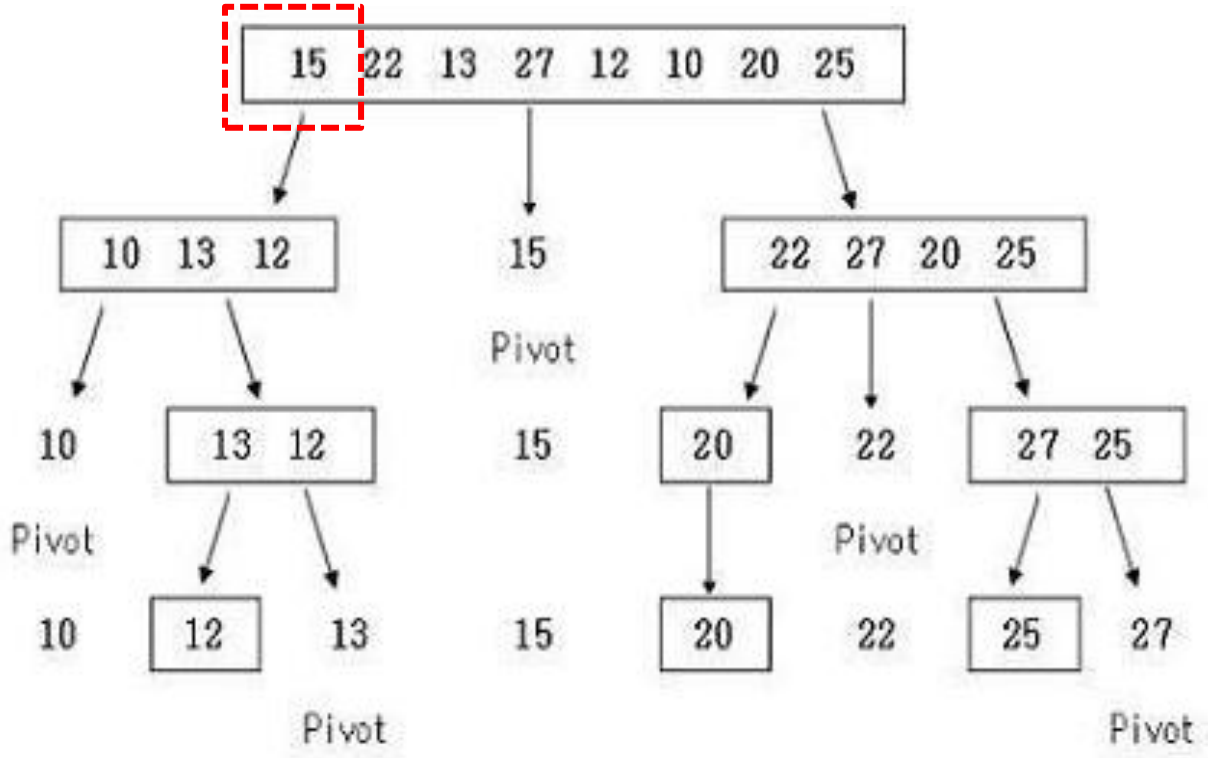
```
void quick_sort(int a[], int n)
    {
    int v, t;
    int i, j;
    if (n > 1)    /* 재귀호출의 !종료 조건 */
        {
        v = a[n-1];  /* v는 축값 */ ①
        i = -1;      /* i는 왼쪽에서부터 검색할 위치 저장 */
        j = n-1;     /* j는 오른쪽에서부터 검색할 위치 저장 */
        while (1)      /* 분할 */
            {
            while (a[++i] < v); /* 왼쪽에서부터 축값보다 큰 값이 있나? */ ②
            while (a[--j] > v); /* 오른쪽서부터 축값보다 작은 값 있나? */
            if (i >= j) break;  /* i와 j의 위치가 뒤바뀌었으면 분할 끝*/
            t = a[i];           /* 아니면 두 값을 바꿈 */ ③
            a[i] = a[j];
            a[j] = t;
            }
        t = a[i];       /* 축값과 축값의 위치에 있는 값을 바꿈 */
        a[i] = a[n-1];
        a[n-1] = t;
        quick_sort(a, i);  /* 왼쪽 소구간에 대해 퀵 정렬 */
        quick_sort(a+i+1, n-i-1); /* 오른쪽 소구간에 대해 퀵 정렬 */
        }
    }
```

5번 사항으로, 'M'을 변수 t에 주고 'O'와 'M'을 t를 통한 SWAP 한다.

⑥ 앞에 인자는 배열 시작 주소, 뒤에는 정렬할 갯수

ITPE Information Technology Professional Engineer

II. 퀵정렬(Quick Sort)을 이용하여 오름차순으로 정렬하는 방법



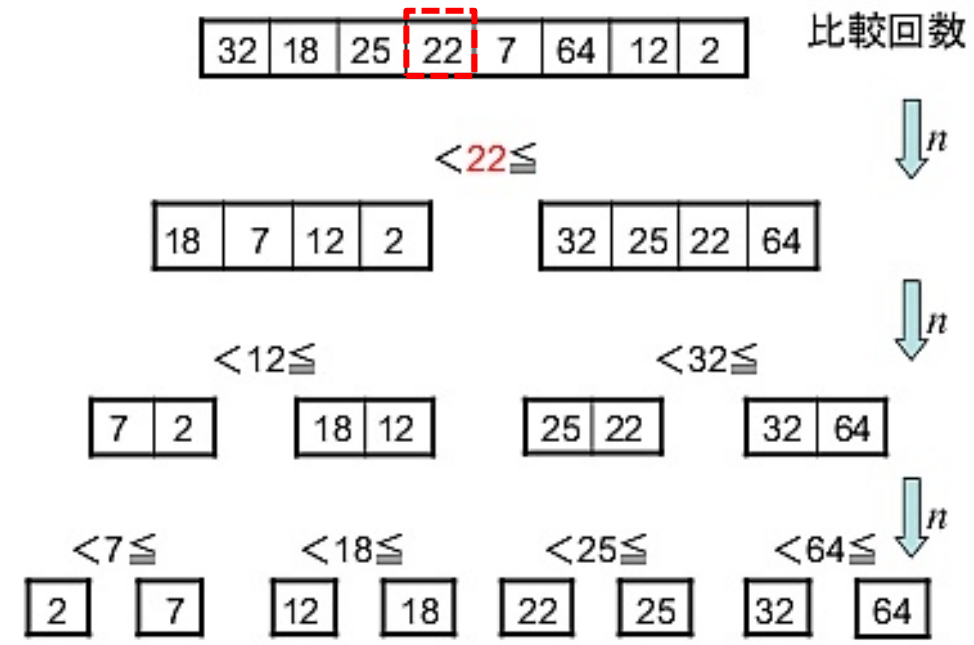
- 축(pivot)값을 기준으로 정렬하는데, 축 값을 중심으로 축 값보다 큰 값은 오른쪽 리스트에 작은 값은 왼쪽리스트로 이동시킨다. (첫번째의 데이터를 축 값으로 한다.)
- 오른쪽 리스트와 왼쪽 리스트부분은 독립적인 단위로 정렬하여 오른쪽 리스트부분에 대한 새로운 분할 축 값을 선택하여 두 부분으로 분리하고, 왼쪽 리스트부분 역시 새로운 축 값을 선택하여 두 부분으로 분리하는 과정을 반복하는데 리스트들은 재귀적 방법으로 각각 재배열 하는 방식.
- 각 분할 자료개수가 1이 되면 정렬은 완료됨

ITPE Information Technology Professional Engineer

III. 사례

- 기준 값을 정해, 나머지 원소들과 비교한 후, 기준 값보다 작은 원소들의 부분, 큰 원소들의 부분 – 두 부분으로 분리하고, 분리된 부분에서도 각각 기준 값을 정한 후에 또 두 부분으로 나누어 정렬하는 알고리즘

(나눈 부분이 1일 될 때까지 반복)



그림의 오른쪽에 나와 있는 한자는 비교횟수라는 의미이다.

ITPE Information Technology Professional Engineer

III. 대표적인 정렬 알고리즘간의 비교

구분	버블정렬	선택 정렬	삽입정렬	퀵정렬
특징	성능 안 좋음. 구현이 단순	비교의 횟수가 많고 교환의 횟수가 적음	비교의 횟수가 적고 교환의 횟수가 많음	피봇(Pivot)
구현	인접한 원소와 비교하여 2개씩 정렬.	정렬되지 않은 부분 에서 최소(최대)값을 찾아 정렬된 부분에 더함.	정렬되지 않은 부분의 값을 정렬된 부분의 적절한 위치로 삽입.	두개의 변수 low와 high 를 사용 1)Low는 피봇보다 작으 면 통과, 크면 정지 2)High는 피봇보다 크면 통과, 작으면 정지 3)정지된 위치의 숫자를 교환: low와 high가 교 차하면 종료
최선	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(n\log_2 n)$
평균	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(n\log_2 n)$
최악(복잡도)	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(n^2)$

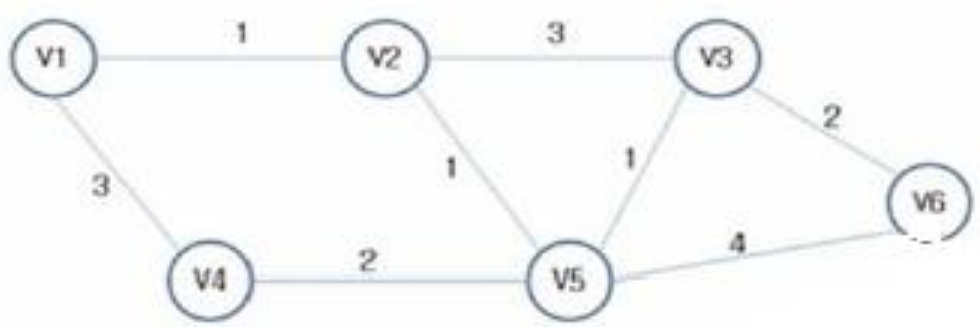
ITPE Information Technology Professional Engineer

7. 그래프(Graph)

출제예상문제

1. 그래프를 이용하여 최적 경로를 찾는 데 이용되는 최소 신장 트리(Minimal spanning tree) 알고리즘에 대하여 설명하시오
(정보관리, 96회 1교시)

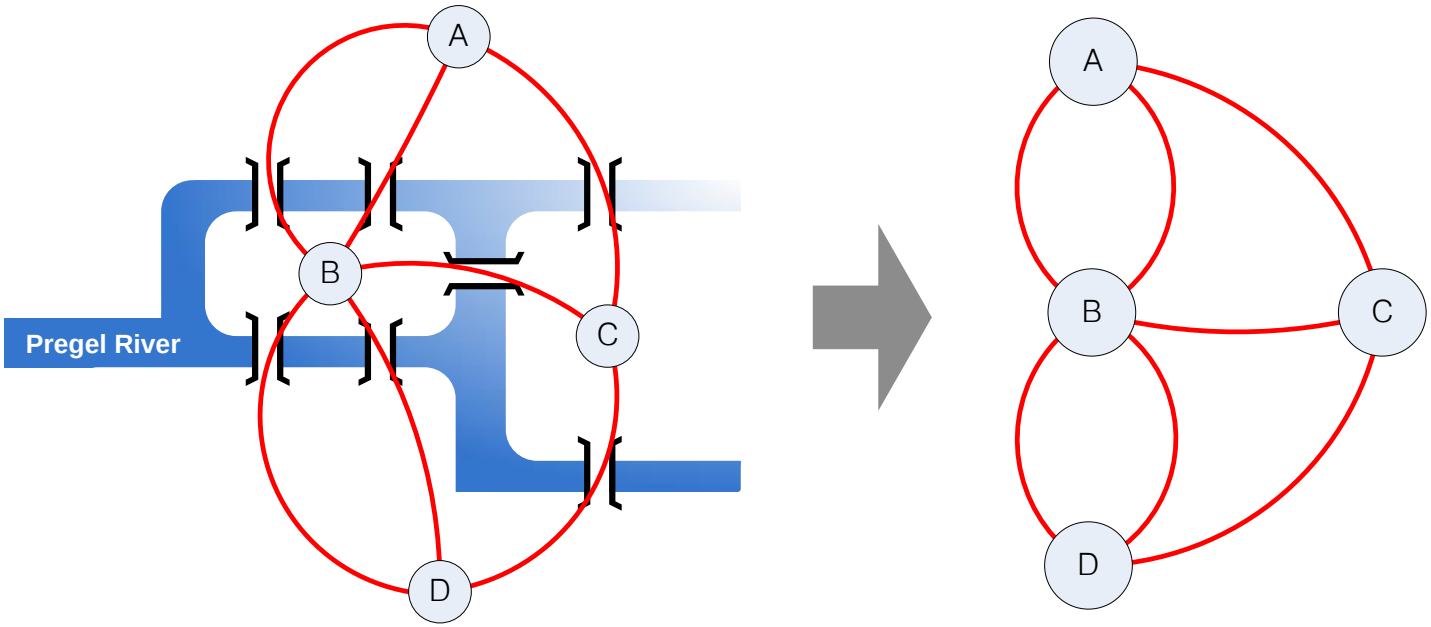
2. 다음 그래프에 Dijkstra가 제안한 최소비용(least-cost)경로설정 알고리즘이 적용되는 과정을 제시하시오. (그래프의 에지옆 숫자는 비용을 의미함)
(정보관리, 84회 3교시 5번)



ITPE Information Technology Professional Engineer

I. 라인하르트 오일러가 “쾨니히스베르크의 7개의 다리 문제”를 풀기 위해 고안해 낸 수학적 도구.

7개의 다리를 간선(Edge)로, 4개의 육지를 정점(Vertex)로 표현.



쾨니히스베르크의 다리 문제는 프로이센의 쾨니히스베르크(지금의 러시아 칼리닌그라드)에 있는 7개의 다리에 관련된 문제이다. 쾨니히스베르크에는 프레겔 강이 흐르고 있고, 이 강에는 두 개의 큰 섬이 있다. 그리고 이 섬들과 도시의 나머지 부분을 연결하는 7개의 다리가 있다. 이때 7개의 다리들을 한 번만 건너면서 처음 시작한 위치로 돌아오는 길이 있는가 하는 것이 문제이다. 1735년에 레온하르트 오일러가 이것이 불가능하다는 것을 증명했다. [그래프에 대한 첫 논문]

ITPE Information Technology Professional Engineer

II. 그래프는 "정점의 모음"과 이 정점을 잇는 "간선의 모음" 간의 결합, 혹은 그림을 말함 (베)

"정점의 집합을 V, 간선의 집합을 E, 그리고 그래프를 G라고 했을 때,

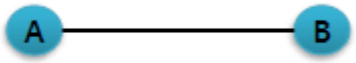
G = (V, E)이다."

정점(V, vertex 또는 node)

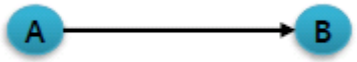
간선(edge)

ITPE Information Technology Professional Engineer

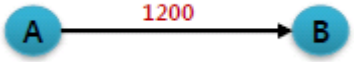
□ 무방향성 그래프(Undirected Graph) : (A, B) = (B, A)

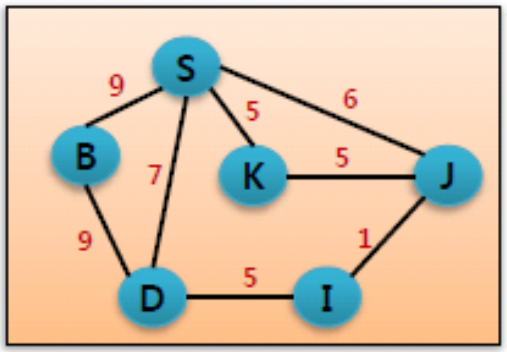


□ 방향성 그래프(Directed Graph) : <A, B> ≠<B, A>

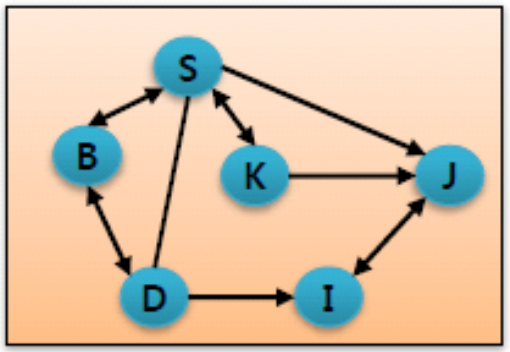


□ 가중치 그래프(Weighted Graph), 네트워크(Network) : 간선에 비용이나 가중치가 할당

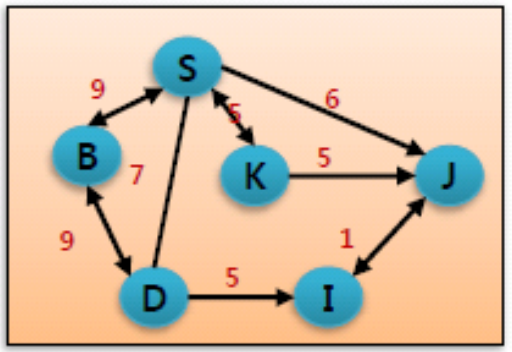




가중치 표시



방향 표시

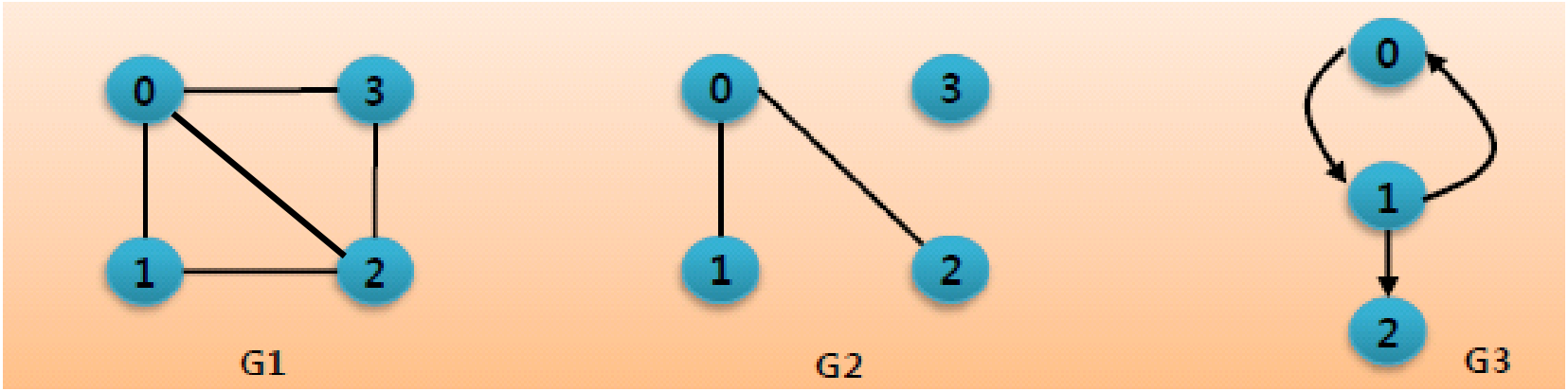


방향/가중치 표시

ITPE Information Technology Professional Engineer

□ 표현의 예 – 무방향성 그래프

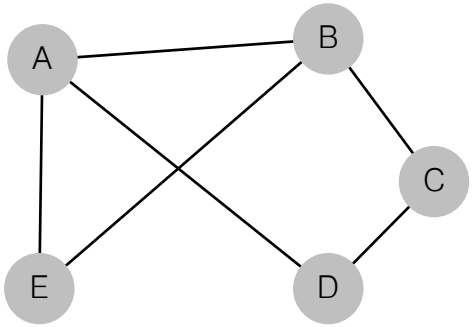
- V(G1) ={0, 1, 2, 3} E(G1) = {(0,1), (0,2), (0,3), (1,2), (2,3)}
- V(G2) ={0, 1, 2, 3} E(G2) = {(0,1), (0,2)}
- V(G3) ={0, 1, 2} E(G3) = {<0,1>, <1,0>, <1,2>}



ITPE Information Technology Professional Engineer

☐ 인접(Adjacent)

- 간선으로 연결되어 있는 두 정점을 일컫는 말. "이웃 관계"에 있다고 표현하기도 한다.



(A, B), (A, D), (A, E), (B, C), (B, E), (C, D)가 서로 이웃 관계

☐ 경로(Path)

- 위 그래프에서 정점 A에서 정점 C까지의 경로는 A-B-C 와 A-D-C, A-E-B-C 등이 있다. A-B-C, A-D-C 경로의 길이는 2, A-E-B-C의 길이는 3이다.

ITPE Information Technology Professional Engineer

□ 깊이 우선 탐색(Depth First Search)

- **"더 나아갈 길이 보이지 않을 때까지 깊이 들어간다." 막다른 길까지!**

1) 시작 정점을 밟은 후 이 정점을 "방문했음"으로 표시합니다.

2) 그리고 이 정점과 이웃하고 있는 정점(즉, 인접 정점) 중에 아직 방문하지 않은 곳을 선택하여 이를 시작 정점으로 삼아 다시 깊이 우선 탐색을 시작합니다. 그러니까 단계 1)을 다시 하는 겁니다.

3) 정점에 더 이상 방문하지 않은 인접 정점이 없으면 이전 정점으로 돌아가서 단계 2)를 수행합니다.

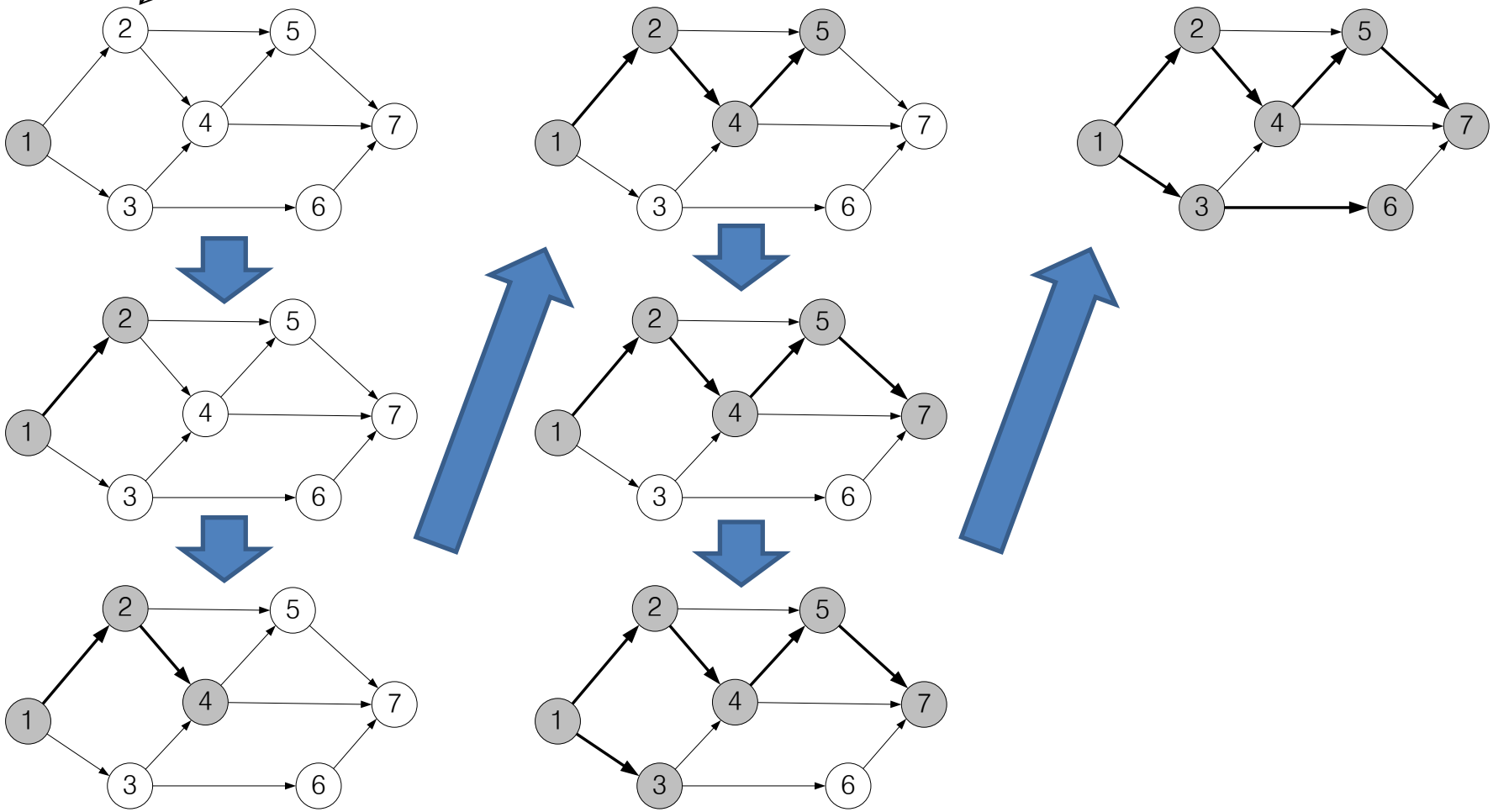
4) 이전 정점으로 돌아가도 더 이상 방문할 이웃이 없다면 그래프의 모든 정점을 모두 방문했다는 뜻입니다. 이제 탐색을 종료합니다.

```
depth_first_search(v)
        v를 방문되었다고 표시;
        for all u ∈ (v에 인접한 정점) do
                if (u가 아직 방문되지 않았으면)
                then depth_first_search(u)
```


ITPE Information Technology Professional Engineer

□ 깊이 우선 탐색의 예

가중치가 작은 정점(Vertex)를 방문



□ 너비 우선 탐색(Breadth First Search)

- "꼼꼼하게 좌우를 살피며 다니자."

1) 시작 정점을 "방문했음"으로 표시하고 큐에 삽입(Enqueue)합니다.

2) 큐로부터 정점을 제거(Dequeue)합니다. 제거한 정점이 이웃하고있는 인접 정점 중 아직 방문하지 않은 곳들을 "방문했음"으로 표시하고 큐에 삽입합니다.

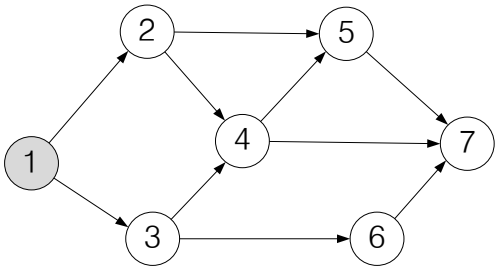
3) 큐가 비게 되면 탐색이 끝난 것입니다. 따라서 큐가 빌 때까지 2의 과정을 반복합니다.

```
breadth_first_search(v)
        v를 방문되었다고 표시;
        큐 Q에 정점 v를 삽입;
        while (not is_empty(Q)) do
            Q에서 정점 W를 삭제;
            for all u ∈ (v에 인접한 정점) do
                if ( u가 아직 방문되지 않았으면)
                    then u를 큐에 삽입하고, u를 방문되었다고 표시;
```


ITPE Information Technology Professional Engineer

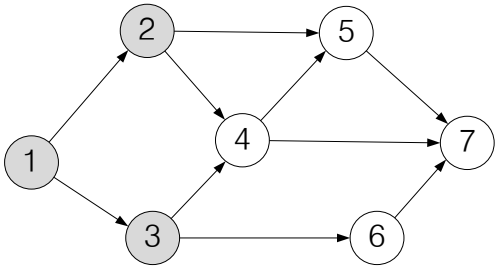
□ 너비 우선 탐색의 예

Queue



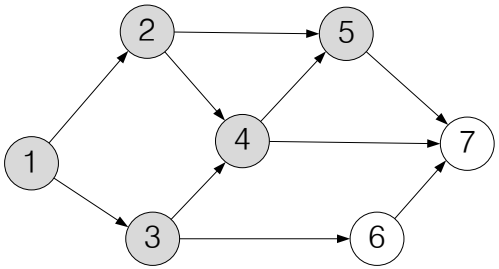
노드1의 첫번째 너비는 2와 3

Queue

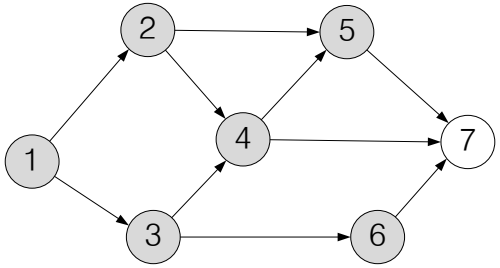


노드2의 첫번째 너비는 4와 5

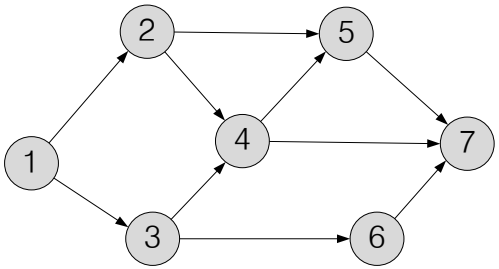
Queue



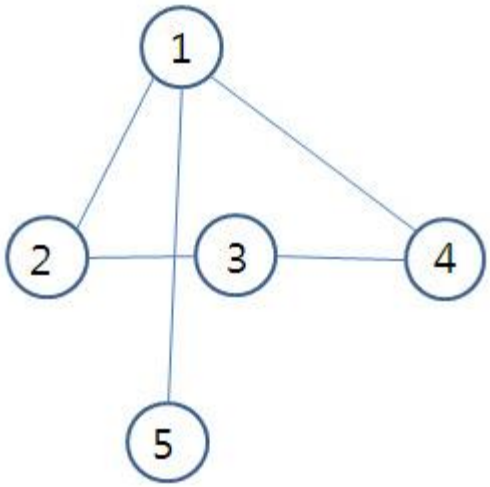
Queue



Queue



[참고]



넓이 우선 탐색(BFS; Breadth First Search)

- 시작 지점이 1이라고 정함.
- 탐색 순위가 넓이 우선 탐색은 시작지점과 갈려고 하는 지점이 같은 것부터 탐색을 시작.
- 즉 출발 지점 1에서 한 칸만 가서 갈수 있는 곳이 2, 4, 5 이다. 여기서 가장 숫자가 적은 2 부터 검색을 하게 된다.
- 다 검색을 한 후에 이제 두 칸을 이동 하여 갈수 잇는 3을 검색 하게 된다.

[결과] 1 , 2 , 4 , 5 , 3 이 순서대로 탐색을 하는 방법.

깊이 우선 탐색(DFS; Depth First Search)

- 출발 지점이 1이라고 정함.
- 가장 가깝고, 가장 적은 노드부터 검색을 시작.

  한 칸 이동한 2로 시작, 2에서 검색 할 수 있는 노드 중에서는 3, 그 다음 4, 이제 모든 검사를 다 한 상태여서, 이전으로 돌아가, 마지막에 5를 검색.

[결과] 1, 2, 3, 4, 5 순서대로 탐색하는 방법

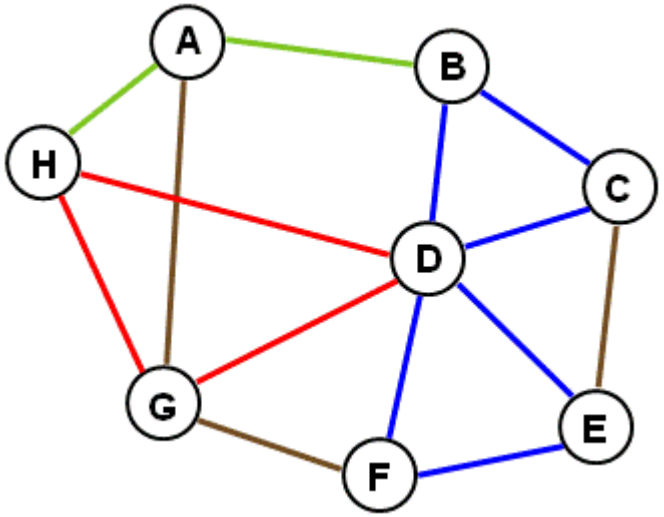
ITPE Information Technology Professional Engineer

8. 최소 신장 트리(Minimum spanning tree)

출제예상문제

1. 출제 예상 문제

[신장 트리]
그래프에서 모든 정점을 포함하고, 정점간 서로 연결되면서, 사이클이 존재하지 않는 그래프



HAB는 시작점과 끝점이 다르므로 순환이 아니다. HDGH는 순환을 이룬다. 다시 자기 노드로 오는 경우를 순환이라고 부른다. [출처] http://ko.wikipedia.org

ITPE Information Technology Professional Engineer

8. 최소 신장 트리(Minimum spanning tree)

출제예상문제

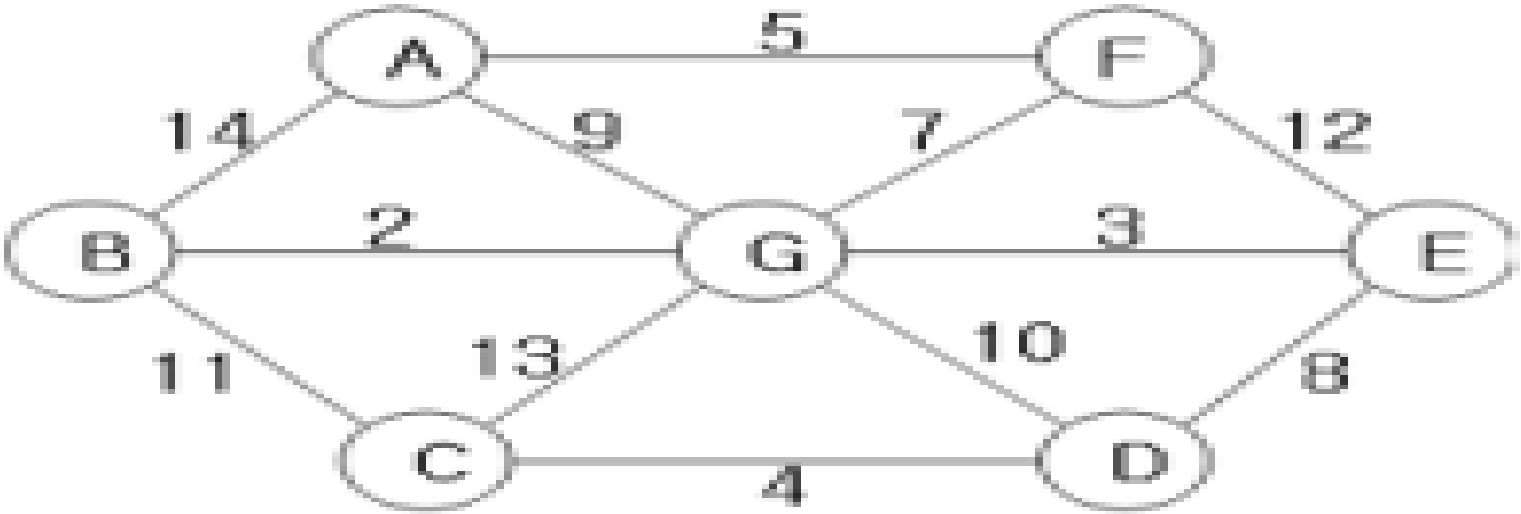
1. 다음과 같이 7개의 신도시의 도로 공사를 최소의 비용으로 설계할 때, 다음 물음에 답하시오.
   단, 노드는 도시 이름을 나타내고 간선은 공사비용이다

가. MST (Minimum Spanning Tree)의 개념에 대해 설명하시오.

나. 위 그래프의 비용인접리스트를 도식화하시오.

다. Prim 알고리즘을 이용해서 MST를 구하는 절차를 보이시오.

라. Kruskal 알고리즘을 이용해서 MST를 구하는 절차를 보이시오.



ITPE Information Technology Professional Engineer

I. 최소 신장 트리(Minimum spanning tree)

간선에 "가중치(Weight)" 속성을 부여, 각 간선이 갖고 있는 가중치의 합이 최소가 되는 신장 트리가 바로 최소 신장 트리

- MST 이해를 위한 그래프 구성

구분	상세설명
그래프	그래프는 G=(V, E)의 유한한 개수의 정점(Vertex) 또는 노드(node)들의 집합인 V와 연결선(Edge)라고 불리는 정점들의 쌍들의 집합인 E로 구성된 자료 구조
경로	그래프 G의 한 정점에서 다음 정점으로 간선이 존재하는 정점들의 순서
사이클	처음 정점과 마지막 정점이 같은 경로
차수	그래프 G에서 한 정점에 교차된 연결선들의 수
교차	정점의 순서쌍 (V1, V2)이 있는 연결선이면, 연결선 (V1, V2)는 정점 V1과 V2에 교차되었다고 함.
인접	정점의 순서쌍 (V1, V2)이 E(G)에 있는 연결선이 존재하는 경우 인접이라고 함.
트리	폐로를 형성하지 않고, 연결되어 있으며, 방향성을 가지지 않는 그래프
신장 트리	주어진 그래프 G의 모든 정점을 연결하는 트리

ITPE Information Technology Professional Engineer

II. Prim 알고리즘을 이용한 MST 도식화 절차

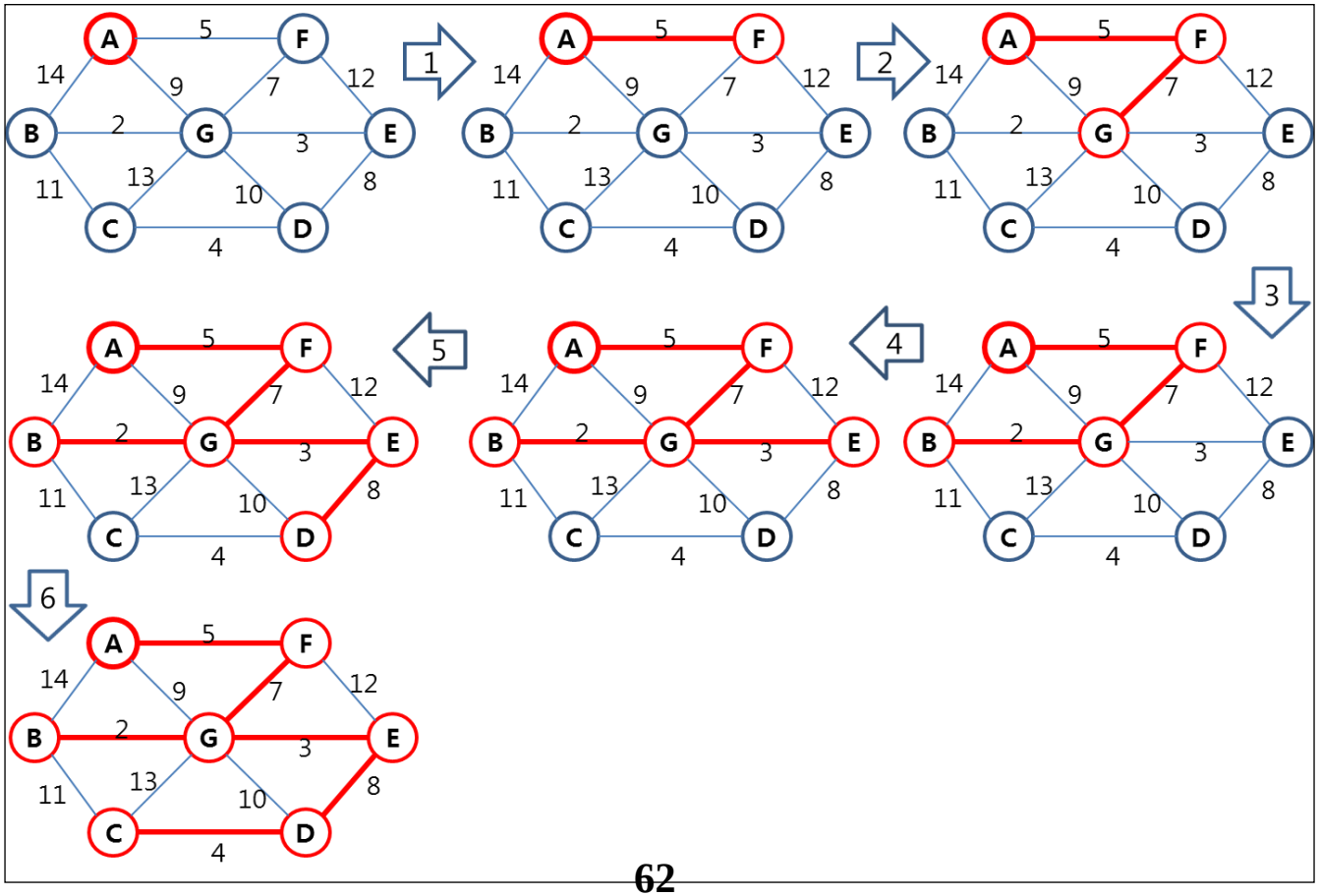
가. Prim 알고리즘의 개념

1) Prim 알고리즘은 시작 정점에서부터 출발하여 신장 트리 집합을 단계적으로 확장해 감.

2) 시작 단계에서는 시작 정점만이 신장 트리 집합에 포함

3) 앞 단계에서 만들어진 신장 트리 집합에 인접한 정점들 중에서 최저 간선으로 연결된 정점을 선택하면서 트리를 확장함.

4) 과정의 종료는 트리가 'n-1'의 간선을 가질 때 종료됨.



ITPE Information Technology Professional Engineer

[설명]

1) 시작점 'A'에서 3개의 node중에 가장 적은 5 가중치로 연결된 'F' 선택

2) 선택된 간선들로 두 정점을 연결했을 때 사이클이 생기면 간선을 포기하고, 사이클이 생기지 않으면 신장트리에 추가. F'와 이미 연결된 5를 제외한 'G'와 'E' 중에서 적은 7의 가중치인 'G' 선택

3) 선택된 간선들 중에서 가중치가 가장 적은 간선들을 택함. 'A', 'F', 'G'에서 사이클을 형성하지 않으며 가장 적은 2 가중치인 'B" 선택

4) 4개의 이미 선택된 Vertex에서 남은 간선들 중에 가장 적은 3의 가중치를 가지는 'G'와 'E'를 연결하는 간선 선택

5) 검사한 간선들 중에서 아직 트리에 들어있지 않으면서, 사이클을 형성하지 않는 간선 중 가중치를 가장 적게 가지는 'D' 선택

6) 남은 간선중 사이클 형성하지 않고, 가중치를 가장 적게 가지는 'C' 선택, 간선 리스트에 더 이상 남은 간선이 없는 'n-1'개의 간선이 선택되어 MST 선정절차 종료.

ITPE Information Technology Professional Engineer

III. Kruskal 알고리즘을 이용한 MST 도식화 절차

가. Kruskal 알고리즘의 개념

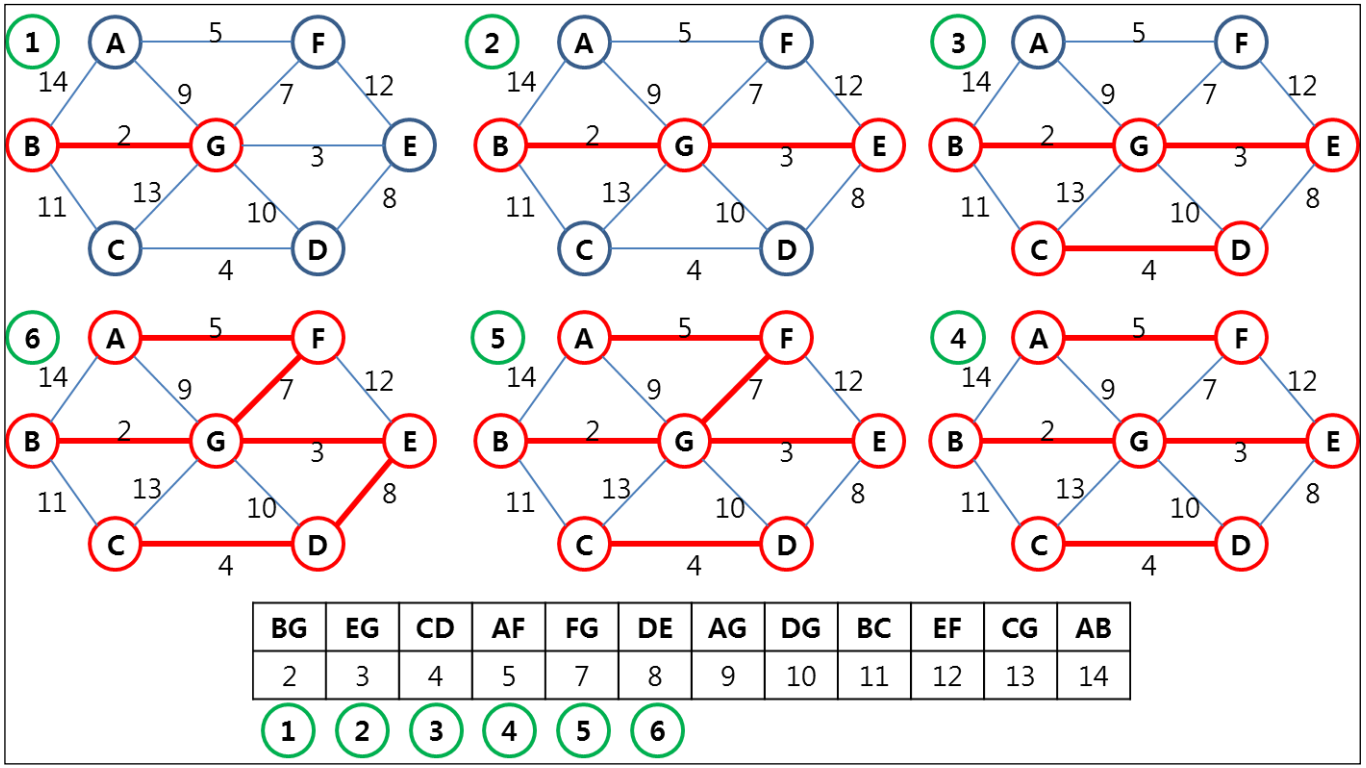
1) MST가 최소의 비용의 간선으로 구성됨과 사이클을 형성하지 않는다는 조건에 충실하면서 각 단계에서 사이클을 이루지 않는 최소비용 간선을 선택함

2) 그래프의 간선들을 가중치의 오름차순으로 정렬

3) 정렬된 간선들의 리스트에서 사이클을 형성하지 않는 간선을 찾아서 현재의 MST 리스트 집합에 추가

4) 만약 사이클 형성한다면 그 간선은 제외

ITPE Information Technology Professional Engineer



1) 각 간선들을 가중치의 오름차순으로 정렬

2) 최소 가중치 간선부터 선택

3) 단, 선택 시 사이클이 발생하지 않는 최소 간선을 선택

4) 'n-1'개의 간선이 선택되면 종료

ITPE Information Technology Professional Engineer

9. Dijkstra(다익스트라) 알고리즘

출제예상문제

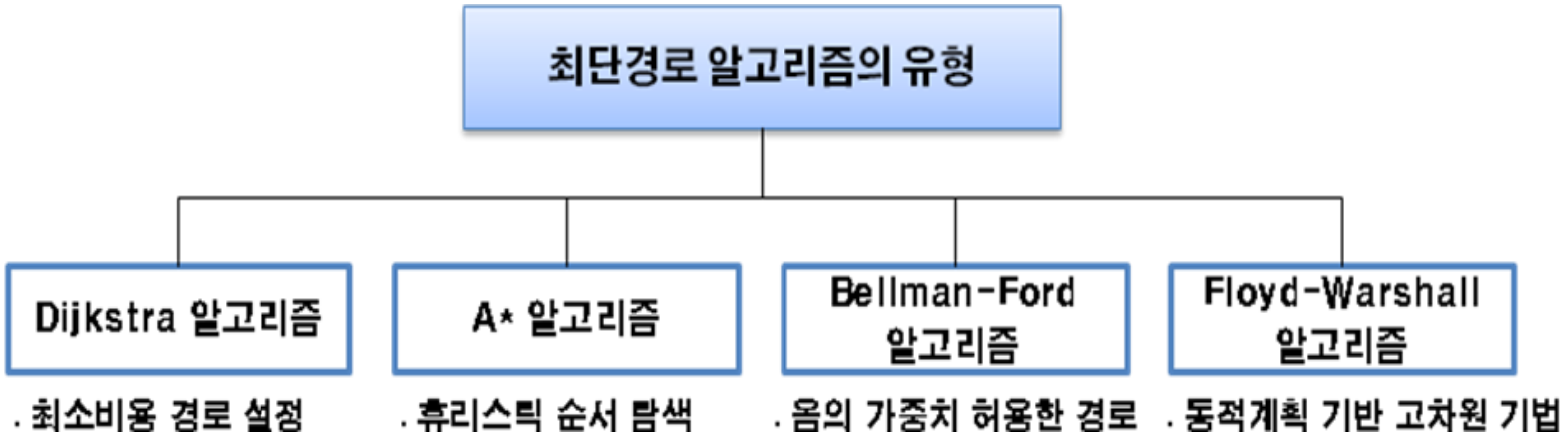
1. 인공지능 응용 정보시스템이나 게임 개발 시 길 찾기에 이용되는 Dijkstra 알고리즘과 A* 알고리즘을 비교 설명하시오.
(정보관리, 98회 4교시)

ITPE Information Technology Professional Engineer

I. 인공지능 시스템과 길 찾기에 이용되는 최단경로 탐색 알고리즘의 개요

가. 최단경로 알고리즘의 유형

- 정점(노드)과 간선(연결하는 중계선)으로 이루어진 그래프에서 두 정점 사이의 가장 짧은 경로
- 가중 그래프에서는 구성하는 구간들의 가중치 합이 최소가 되도록 하는 경로
- 일반적으로 간선에 가중치가 있는 유향(有向 ; 방향이 있는) 그래프에서의 알고리즘



- 최근 인공지능 응용 정보시스템 분야에서 Dijkstra 알고리즘이 많이 활용되며, 게임 개발 시 여러 맵상의 경로를 구하는 데 효과적으로 사용되는 A* ('에이 스타'라고 발음) 알고리즘이 많이 쓰임

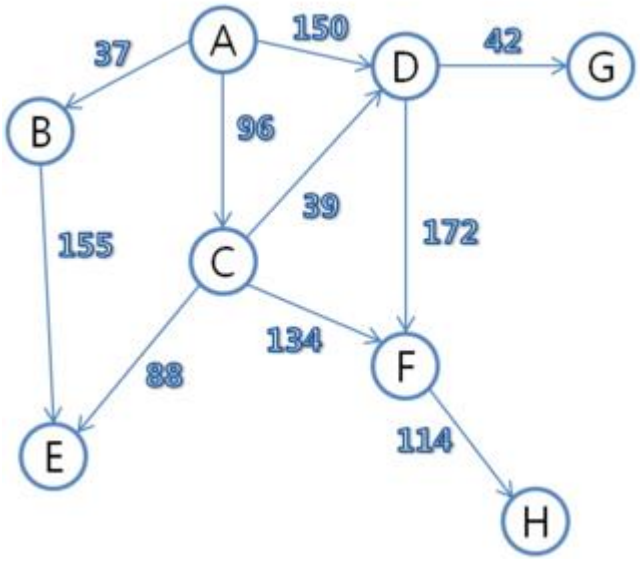
ITPE Information Technology Professional Engineer

나. Dijkstra 알고리즘과 A*알고리즘의 개념 차이

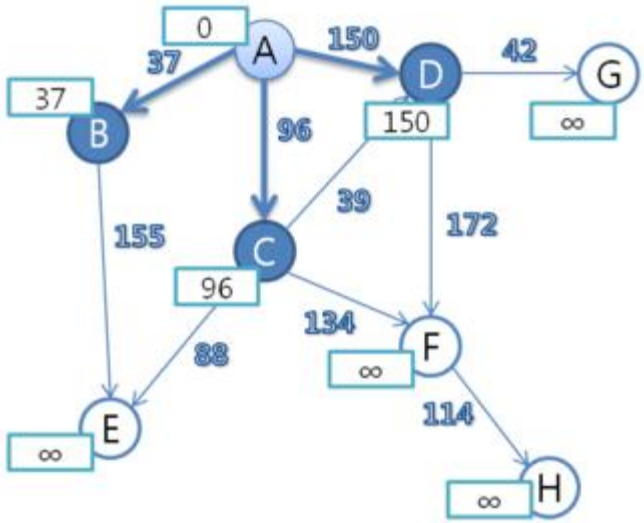
구분	Dijkstra 알고리즘	A* 알고리즘
경로선택	출발점이 주어지면 다른 모든 정점에 이르는 최단 거리를 구함	해당 시점에서 특정영역 이동 시 목적하는 방향을 보고 다음 목적지 선택 - 넓이 우선 탐색에 깊이 우선 탐색을 조합하여 휴리스틱한 기법 사용
	출발점은 주어지나 도착지점은 잘 주어지지 않는 경우 각 정점에서 최단거리 선택	반드시 도착지점이 주어지는 네비게이션에 활용
평가함수/추정거리	평가함수/추정거리를 사용하지 않음	평가함수를 사용하여 추정거리를 계산함
네트워크 경로	모든 경우의 수에 해당하는 라우팅 경로 테이블 구성 시 적용	최적의 응답속도를 위한 라우팅 경로 테이블 구성 시 적용

ITPE Information Technology Professional Engineer

II. Dijkstra 알고리즘 경로 사례

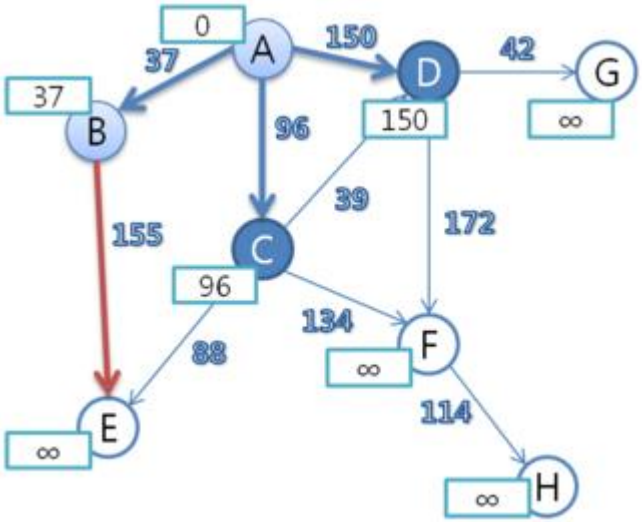


- 위와 같은 그래프가 주어졌을 때, A가 시작점으로 B ~ H 까지의 최단경로 길이를 구하는 것이 목표
- 모두 무한대로 설정.

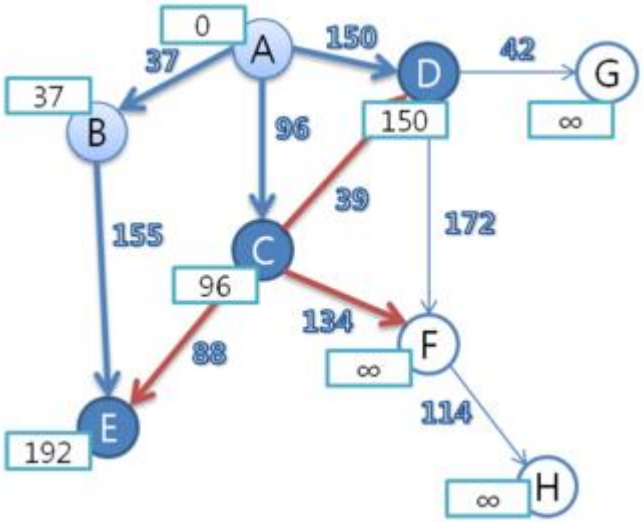


- 시작점 A와 인접한 정점까지의 거리를 계산하면,
- B = 37, C=96, D=150 임.
- 현재까지 최단 경로로 갈 수 있는 상태하고 결정함.

ITPE Information Technology Professional Engineer



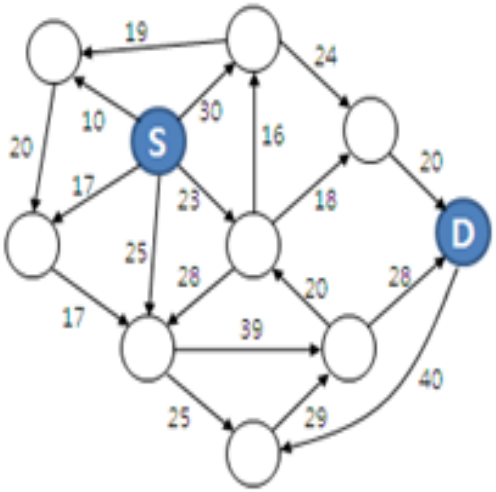
- B = 37, C=96, D=150 임.
- 정점 B를 최단 경로로 확정하고, B를 통한 유일한 경로는 A -> B -> E 가 나오며, 경로 가중치는 192입니다.
- A –> C: 96
- A -> D: 150
- A -> B -> E: 192
- A -> C 거리가 현재 가장 짧으므로, C를 확정하고,
- 새로운 경로 찾기

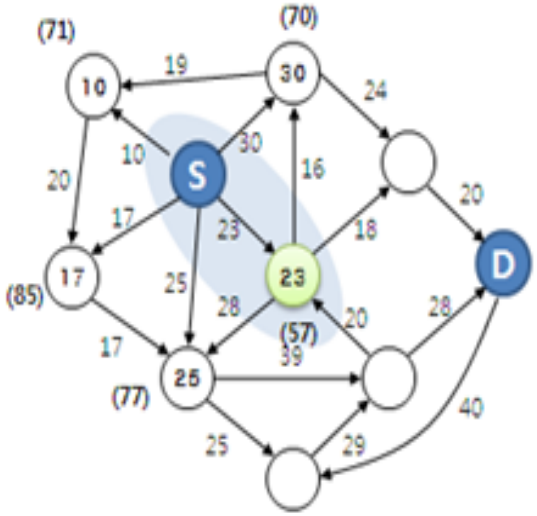


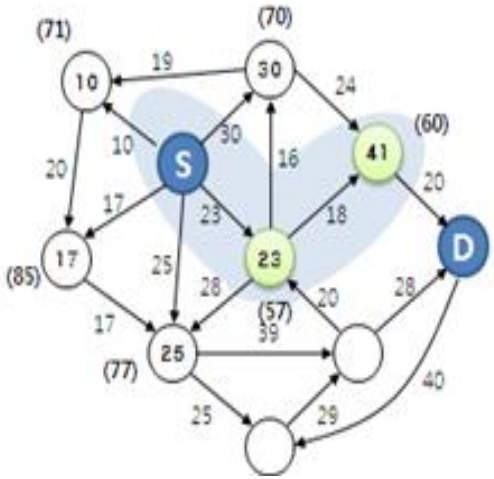
- A -> D (150) 보다 A -> C -> D 거리가 135로 더 짧으므로 갱신합니다.
- A -> B -> E(192) 보다 A -> C -> E 거리가 184로 더 짧으므로 갱신합니다.
- A -> C -> F는 230으로 갱신합니다.
- A -> C -> D: 135
- A -> C -> E: 184
- A -> C -> F: 230
- 이 중에서 A -> C -> D: 135 가장 짧으므로, D를 확정하고 새로운 경로를 봅니다.

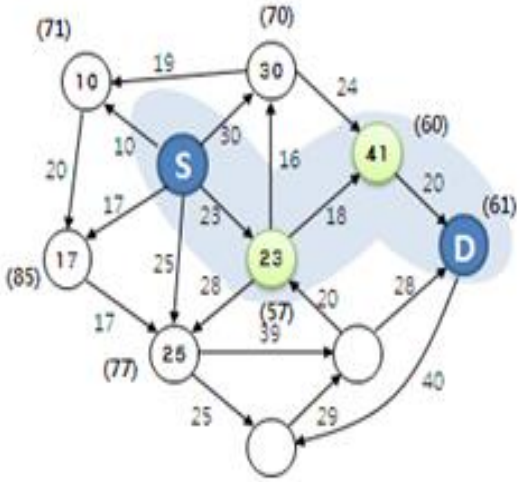
ITPE Information Technology Professional Engineer

III. A* 알고리즘









- Dijkstra 알고리즘과 달리 추정잔여거리를 사용함으로써 탐색의 단계가 감소

- Dijkstra 알고리즘 보다 빠른 탐색이 가능

ITPE Information Technology Professional Engineer

III. 인공지능 시스템과 길 찾기에 사용되는 Dijkstra 알고리즘과 A* 알고리즘 일반사항 비교

가. Dijkstra 알고리즘과 A* 알고리즘의 개념 및 탐색원리 비교

항목	Dijkstra 알고리즘	A* 알고리즘
개념	- 네트워크에서 하나의 시작 정점으로부터 모든 다른 정점까지의 최단 경로를 찾는 알고리즘 - 항상 가까운 거리를 갖는 정점으로의 에지를 선택하여 인접 정점에 대해 오직 하나의 후보 에지만 추적	- 그래프의 시작점에서 도착점에 이르는 최단 경로를 찾는 알고리즘으로 최적 우선 탐색 기법을 이용하는 알고리즘 - 현재까지 계산해 놓은 노드의 평가함수와 목적점에 이르는 잔여 비용의 추정치를 합한 수치를 기준으로 해당 노드의 선택 여부를 결정 하는 알고리즘
탐색원리	- 시작정점과 최초 연결한 근접 정점을 집합 S로 규정 - 집합 S에 포함되지 않은 정점 중에서 시작 정점으로부터 가장 가까운 정점을 탐색 (집합 S에 이웃한 정점 중 하나) - 결정된 정점을 집합 S로 포함 - 새로운 정점이 없어질 때까지 반복 - 모든 정점이 집합 S에 포함되었다면 종료	- 시작 정점으로부터 목표정점 방향의 가장 근접한 정점을 선택 (휴리스틱 함수사용) - 이미 조사된 정점을 닫힌 목록으로, 조사 대상의 정점을 열린 목록로 규정 - 열린목록에서 가장 유망한 정점을 취하며 조사된 정점은 닫힌 목록으로 소속 - 목표정점에 도달할 때까지 열린 목록을 조사하며 닫힘 목록으로 변환시키는 작업을 반복 수행

나. Dijkstra 알고리즘과 A* 알고리즘의 조건과 특징 비교

항목	Dijkstra 알고리즘	A* 알고리즘
조건	- 시작점 있으면 모든 정점의 최단거리 구함 - 가중치가 주어진 유향그래프 - 음의 가중치가 있는 경우 사용 불가	- 최초 시작 정점이 주어짐 - 최단경로 보장 (단, h*(n) <= h(n)일 경우) - 알고리즘 평가함수 사용
특징	- Greedy 알고리즘 - 무조건 최소비용 경로만 선택 - 최적의 해를 증명하지 못함 - 한번 계산된 정점은 다시 계산되지 않음 [참고] 그리디 알고리즘(Greedy Algorithm)에서의 그리디의 뜻은 탐욕,욕심쟁이의 뜻 가짐. 즉, 설계 기본방침은 선택할 것이 여러 개일 경우 그 중 가장 좋은 것을 취한다는 것 (욕심쟁이 방법) [예] Dijkstra의 Shortest Path Algorithm(최단경로 알고리즘) 이나 MST(Minimum Spanning Tree = 최소신장트리) 같은 유명한 알고리즘들은 모두 그리디를 이용한 알고리즘.	- 휴리스틱 함수를 활용 - 모든 정점이 아닌 목표 정점까지의 최단거리 탐색 - 최적경로 탐색을 위한 평가함수 사용 - 시스템의 자원을 가장 적게 사용

다. Dijkstra 알고리즘과 A* 알고리즘 활용분야와 고려사항 비교

항목	Dijkstra 알고리즘	A* 알고리즘
활용분야	- 네트워크 라우팅경로 설계 : OSPF(Open Shortest Path First) 라우팅 프로토콜 활용 - 지능형 교통분야 : GIS, 교통정보, 계획/설계분야 : 도시계획, 철도 및 도로건설, 항공기 운항 등의 분야에 활용	- 네비게이션 : 목적지까지의 최단 경로제공을 통한 운전자 운행정보 제공 - 전략시뮬레이션 게임에서의 유니트 길찾기 : 게임 유닛에 대한 목표지점 지정시 알고리즘을 통한 최단 경로를 활용한 이동
고려사항	- 최선경로여부 : 최상의 해법이 아닐 수 있음. - 수행시간분석 : 입력자료의 상태에 따라 수행시간의 편차가 큼 (해결책: 알고리즘 평가시 최악,평균,최선 등으로 구분하여 수행능력을 분석)	- 컴퓨팅 리소스 낭비 : 맵의 크기가 큰 경우 '열린'목록과 '닫힌'목록의 수가 급격히 늘어나 메모리 부족 (해결책: 깊이우선 탐색의 장점인 메모리 효율성을 탐색방법에 적용하여 구현) - 경로 존재 여부 사전 파악 불가 : 경로가 존재 않을 시 맵을 미리 분석하여 정보 파악

ITPE Information Technology Professional Engineer

Hash

기출문제

1. 정적해싱(Static Hashing) 과정에서 발생하는 오버플로우(Overflow)를 처리하기 위한 전형적인 기법 2 가지를 제시하고, 성능 관점에서 비교하여 설명하시오.
(104회 정보관리기술사, 4교시 4번)

2. 해쉬 테이블(Hash Table)의 개념과 장단점, 활용분야 및 충돌 해결(Collision Resolution)의 여러 가지 기법에 대하여 설명하시오.
(컴시응, 98회, 4교시)

3. Hashing에 대해 설명하시오. (모의고사)

4. 해싱의 개요 및 해시함수의 종류와 해시충돌 해결방법에 대해 기술하시오. (모의고사)

5. 해시 테이블에 대해 다음 질문에 대하여 설명하시오.
가. 해시 테이블 자료구조의 정의
나. 해시 함수의 기법
다. 해시 충돌 해결방안 (모의고사)

ITPE Information Technology Professional Engineer

I. Hashing의 개념

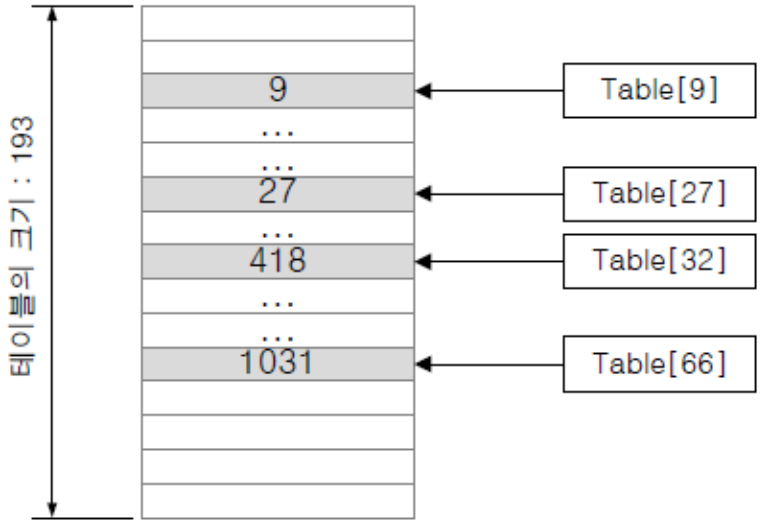
- 주어진 속성값을 기초로 하여 원하는 목표 레코드를 직접 접근할 수 있게 하는 기법
- 키값에 직접 산술적인 연산을 적용하여 레코드가 저장되어 있는 테이블 주소를 계산 (Key-to-Address conversion) 하며, 이렇게 키값에 의한 탐색이 가능한 구조를 해시테이블이라 부르며 해시테이블을 이용한 탐색방법

II. 개념도와 구성요소는 보안 도메인 참조 요망

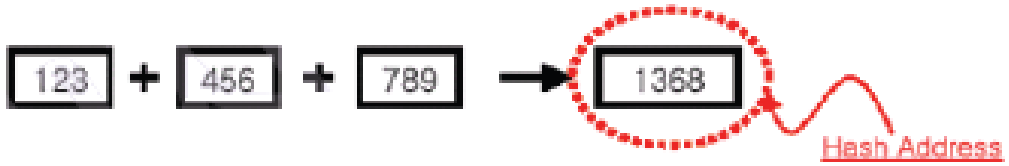
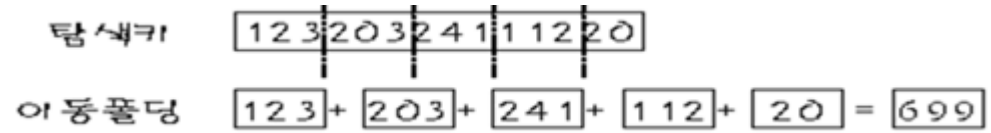
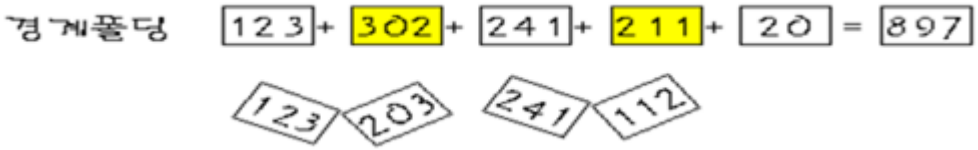
ITPE Information Technology Professional Engineer

III. Hash Function의 종류(혹은 기법으로 시험 문제가 나옴)

가. 나눗셈법(Division)

함수	내용	방법 , 예제
나눗셈법 (Division)	h(key) = key % M (M : 버킷의 크기, 소수) 정수인 탐색키(입력값)를 버킷의 크기로 나눈 나머지를 해시 주소로 사용 [정리] %를 적용한 함수는 mod 함수이다.  	해시 주소의 고른 분포 보장 힘듬 → M은 소수(Prime Number)로 선택 [정리] prime number(= 소수)의 의미는, 양의 약수가 1과 자기 자신 뿐인 1보다 큰 자연수

나. 폴딩법(Folding)

폴딩법 (Folding)	키값(입력값)을 레코드의 키를 마지막 부분을 제외한 모든 부분의 길이가 동일하게 여러 부분으로 나누고, 이들 부분을 모두 더하거나 배타적 논리합(XOR)을 취하여, 해쉬 테이블의 주소로 이용하는 방법.  다시, 이것이 2개로 분류됨 1) 이동 폴딩(Shift Folding): 수를 더함  2) 경계 폴딩(Boundary Folding): 이웃한 부분의 수를 뒤집어서 더함  	123456789012을 3개로나눔 1) 이동폴딩 : 123+456+789+012 2) 경계폴딩 : 123+654+789+210

ITPE Information Technology Professional Engineer

다. 중간 제곱(Mid Square)

중간제곱	레코드 키 값을 제곱한 후 결과값의 중간 위치의 비트들을 선택하여 해시테이블의 홈 주소로 결정 예) h(265) = 70225의 중간값 = 022 위 265를 제곱함. 제곱한 중간값인 022의 비트값을 아래와 같이 표현하면, 비트로 표현이 가능하다. 중간 4비트 혹은 6비트를 선택한다. 	옆의 사례로 대치함.

라. 기수변환법((Radix-Exchange)

기수변환법	주어진 레코드 키를 특정한 진법의 수로 간주하고 키를 변환하여 홈 주소를 얻는 방법 예) 키의 항목이 10진수 [1234]로 되어 있을 경우에 7을 기수로 하여 변화시킬 때 상대주소는[466]이 된다. 1*7^3 + 2*7^2 + 3*7^1 + 4*7^0 (^는 승을 말함)

위 4가지는 암기하고, 그 외에 자릿수 분석법 (Digit-Analysis), 무작위방법((Pseudo-Random)이 있다.

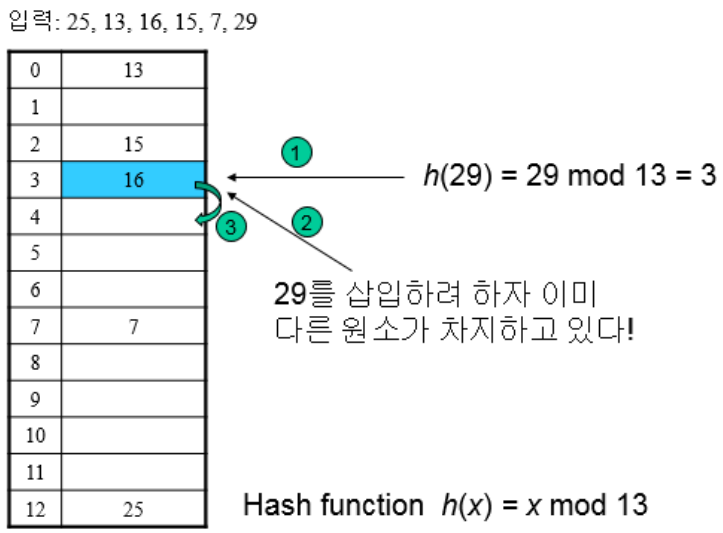
참고주소: http://cafe.naver.com/81th/96966 김영상님 자료

ITPE Information Technology Professional Engineer

IV. 해시 충돌 해결방안 - 오버플로우가 일어났을 때 해결방법

가. Open Addressing

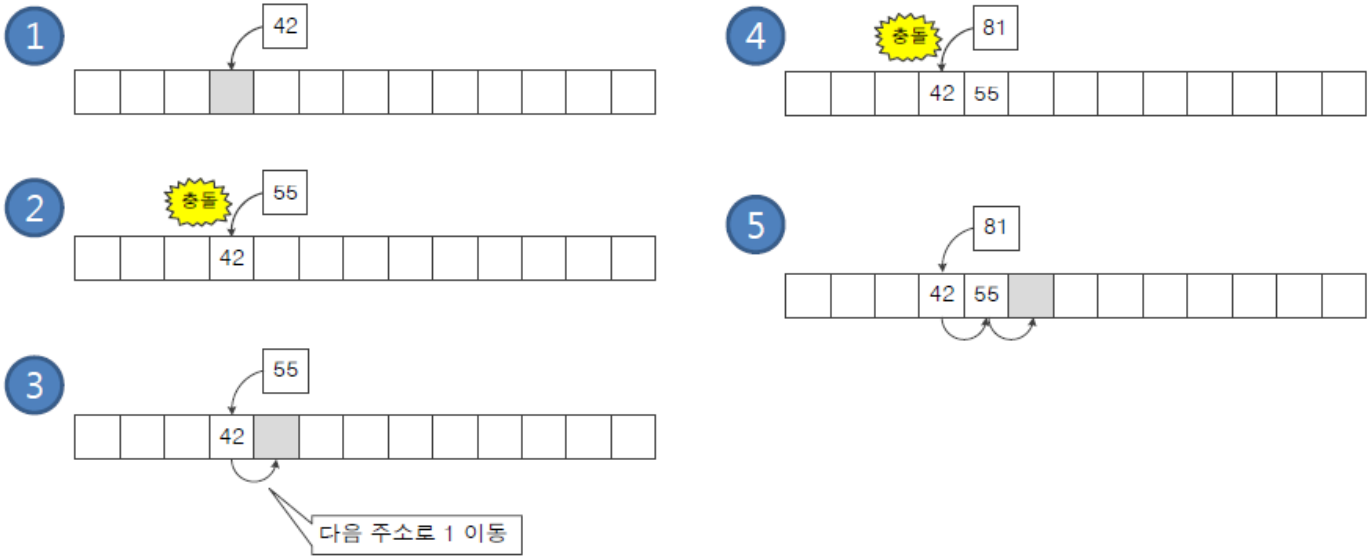
- 개방 주소법(Open Addressing) : 오버플로우가 일어났을 때 다른 데이터 주소로 다시 해싱 시키는 방법

기법	개념도	설명
Open Addressing		- 설명: 충돌(오버플로우)발생 시 빈자리를 탐색하여 해시 값을 저장하는 방식 [해결 방법/종류] - 선형조사법 (Linear probing) - 이차조사법 (Quadratic probing) - 이중해싱법 (Double hashing probing) - 무작위검색 (Random Probing) · Probing(프로빙): 조사, 면밀히 살피는 (기법)

ITPE Information Technology Professional Engineer

1) 선형조사법 (Linear Probing)

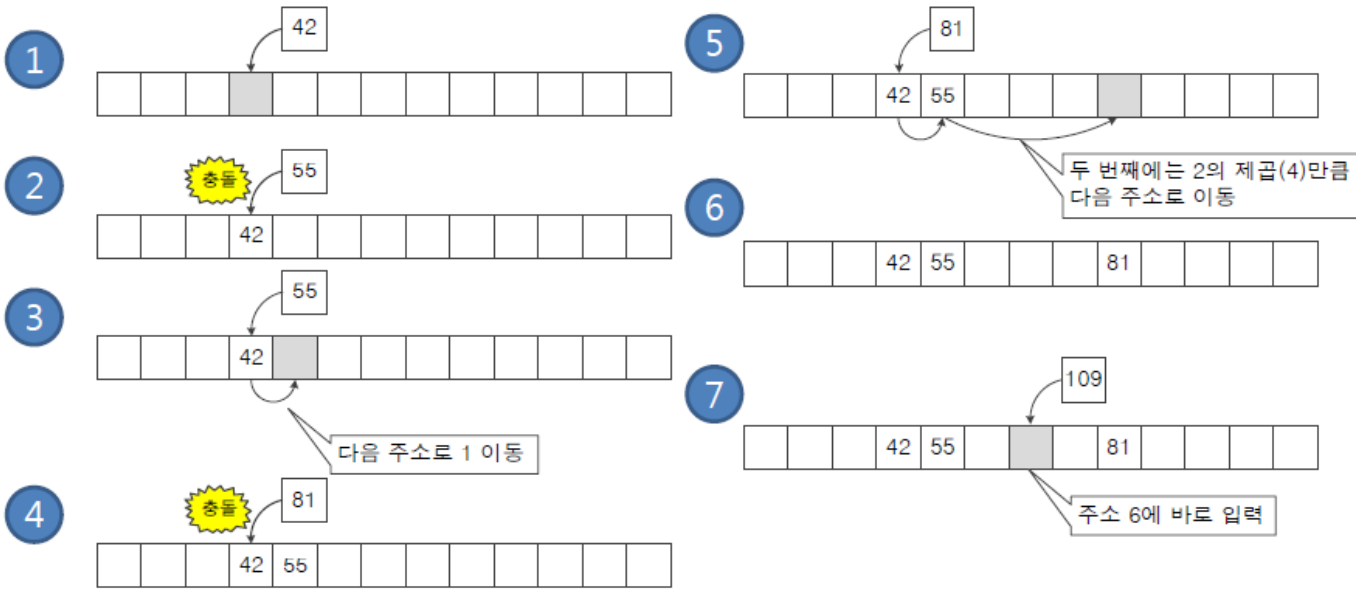
기법	상세설명
선형조사법 (Linear Probing)	순차적으로 탐색하며 비어있는 버켓을 찾을 때까지 계속 진행 최악의 경우, 탐색을 시작한 위치까지 돌아오게 되어 종료(모든 버켓 순회) 캐시의 효율은 높으나 데이터의 클러스터링에 가장 취약한 단점이 있음. [설명] 해시 함수로부터 얻어낸 주소에 이미 다른 데이터가 입력되어 있음을 발견하면, 현재 주소에서 고정 폭(예를 들면 1)으로 다음 주소로 이동 • 충돌이 ht[k]에서 충돌이 발생했다면 ht[k+1]이 비어 있는지를 조사 만약 비어있지 않다면 ht[k+2]를 살펴본다



ITPE Information Technology Professional Engineer

2) 이차조사법(Quadratic Probing)

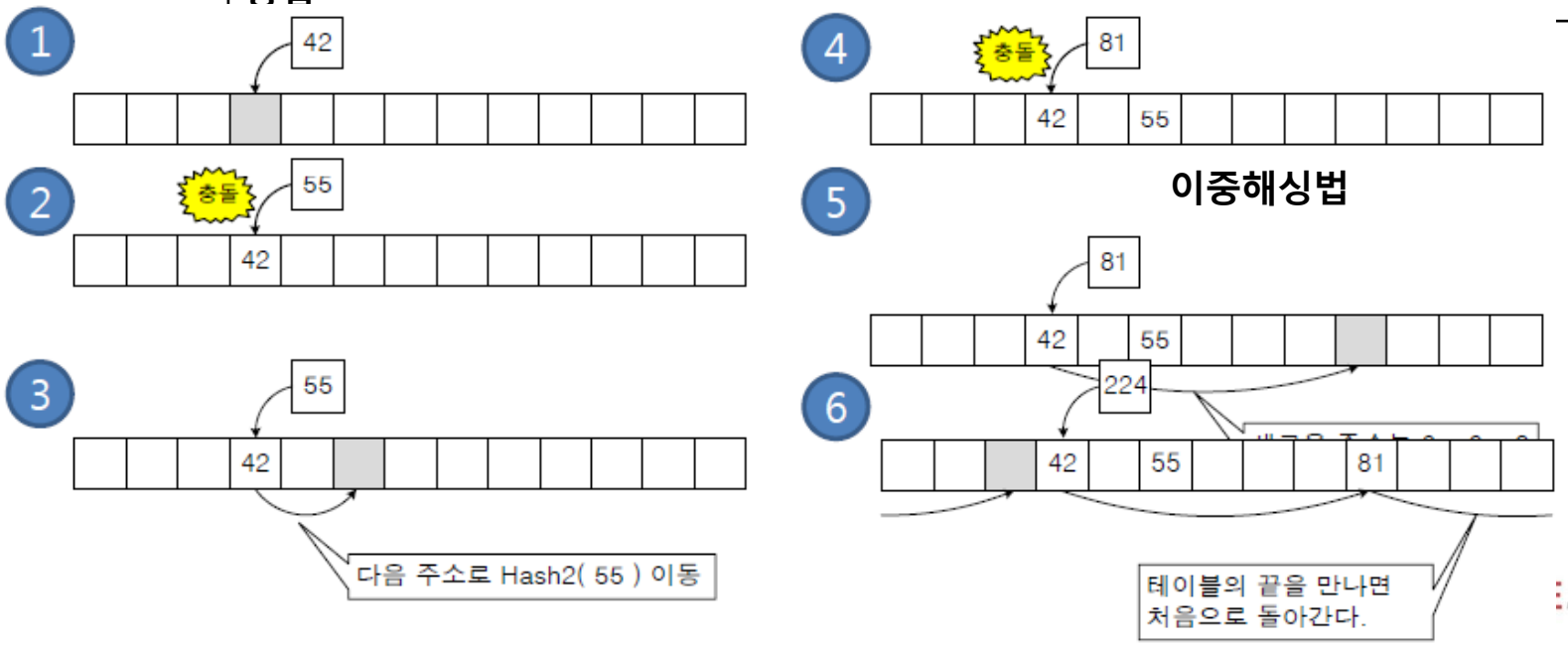
기법	상세설명
이차조사법 (Quadratic Probing)	2차 함수를 이용해 탐색할 위치는 찾음. 캐시의 효율과 클러스터링에 강한 능력에서 linear probing과 double hashing probing의 중간 정도 능력을 가짐. • Quadratic: 2차의 예) 원래의 테이블 주소로부터 다음 주소를 결정하는 거리 d가 1, 4, 9, 16, ...과 같이 떨어진 영역을 차례대로 검색하여 처음으로 발견되는 빈 영역에 해당레코드를 저장함으로써 충돌을 해결하는 방법이다.



ITPE Information Technology Professional Engineer

3) 이중해싱법(Double Hashing Probing)

기법	상세설명
이중해싱법 (Double hashing Probing)	하나의 해시 함수에 의해 충돌이 발생하면 2차 해시 함수를 이용해 새로운 주소를 할당하는 방법. 2개의 해시 함수를 준비해서 하나는 최초의 주소를 얻을 때 또 다른 하나는 충돌이 일어났을 때 탐사 이동폭을 얻기 위해 사용 가장 많은 연산량을 요구.
무작위검색 (Random Probing)	충돌을 유발한 레코드 키를 저장할 수 있는 가용공간을 찾을 때까지 난수 계산 프로그램을 실행하여 해시 Table(hash table)의 홈 주소를 다음 후속 주소로 택하여 충돌을 해결하는 방법

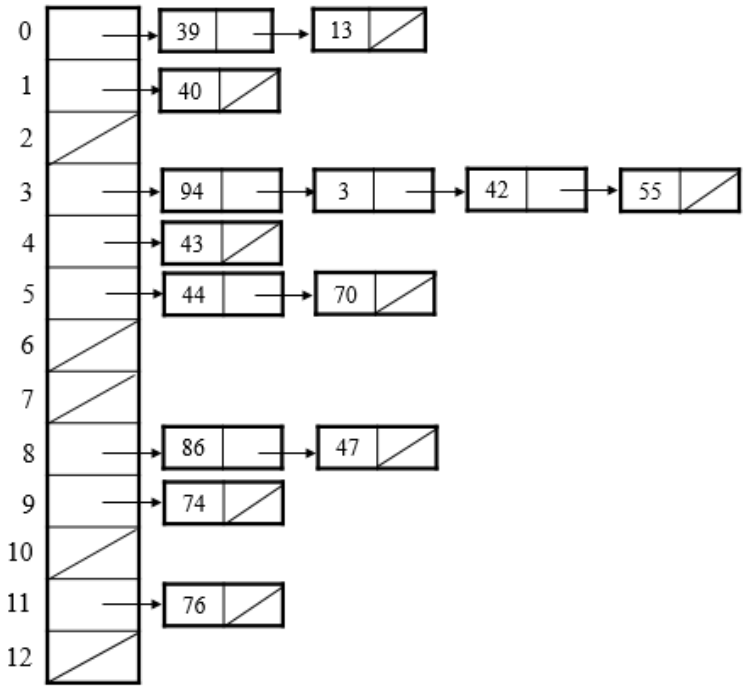
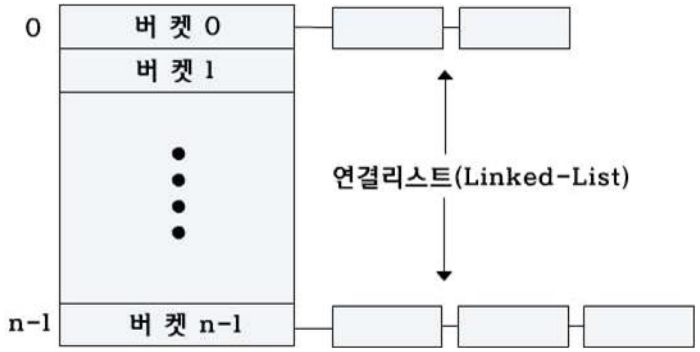


나. Closed addressing(폐쇄 주소법, 혹은 체인법 이라고 부름)

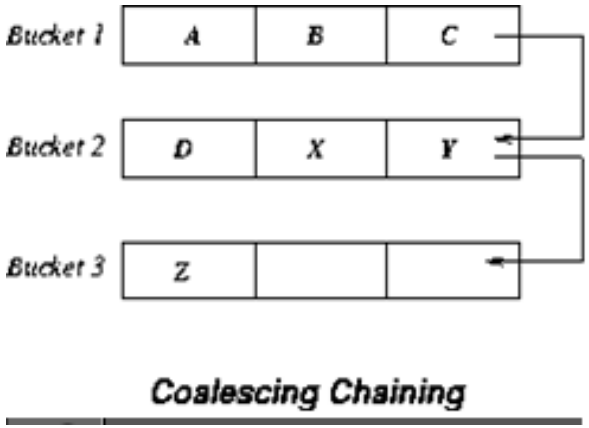
- 같은 데이터 주소 내에서 끝까지 해결을 보는 방법으로 이 방법은 같은 해쉬값을 가지는 모든 레코드들을 리스트(LIST)로 만들어 관리한다. (아래 그림을 개념도로 사용하면 됨)

- Chaining

1) 해쉬 체이닝 (Hash Chaining)

기법	개념도	설명
Hash Chaining		- 설명: 충돌(오버플로우) 문제를 연결 리스트로 해결. - 각 버켓에 고정된 슬롯을 할당하는 것이 아니라 각 버켓에, 삽입과 삭제가 용이한 연결 리스트를 할당 

2) Coalesced Hashing (or Chaining with Coalescing Lists)

기법	개념도	설명
Coalesced Hashing	 Coalescing Chaining Coalesce: 코올레스, 합치다 라는 의미. 포인트가 있어야 함.	- 이 방법은 보통 버킷(bucket)과 함께 사용되곤 하는데, 한 버킷에서 오버플로우(overflow)가 일어나게 되면 그 다음에 나오는 빈 버킷을 찾아서 그곳에 오버플로우 된 레코드를 집어 넣고, 그쪽과 원래 해쉬되었던 버킷을 포인터로 연결시키는 방법이다. - 따라서 해쉬값이 다른 레코드들이 같은 버킷 내에 존재하는 것이 가능하다.

ITPE Information Technology Professional Engineer

[결론]

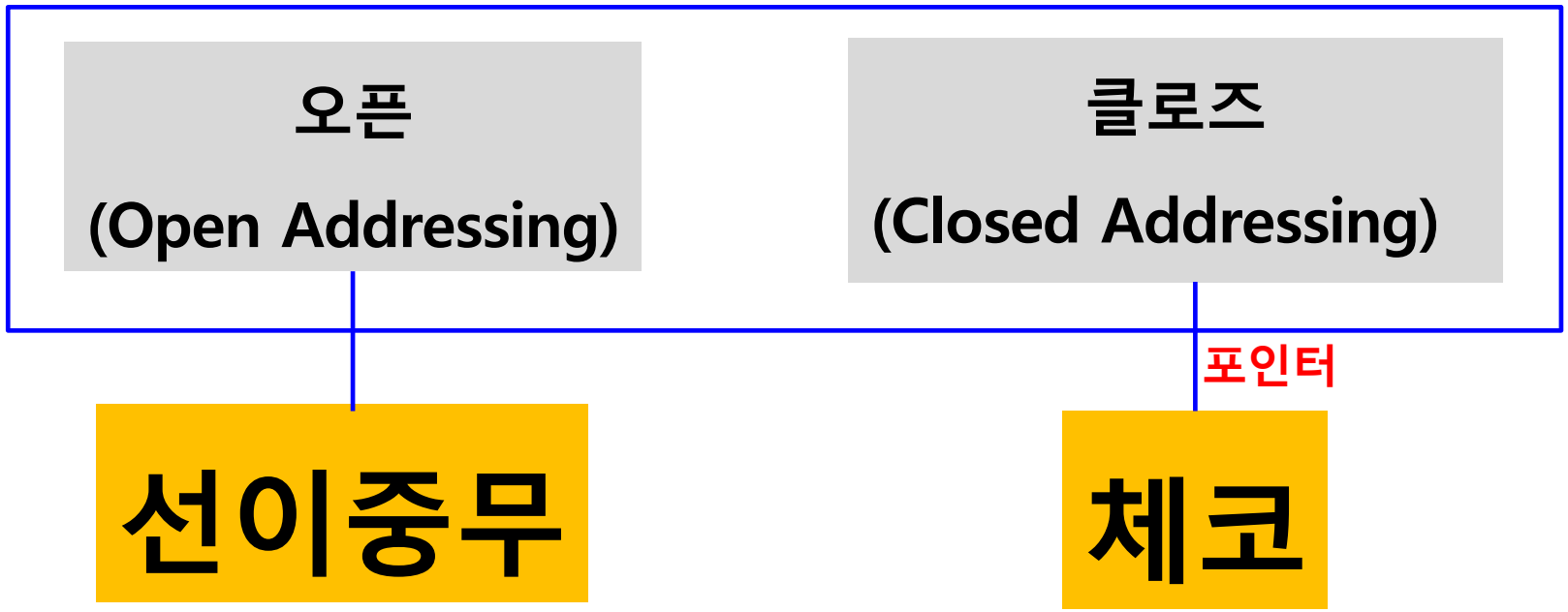
Open Addressing 장점	Chaining 장점
● Open-Addressing 방식은 어떠한 포인터도 저장할 필요가 없고 테이블 외부에 어떠한 추가적인 저장 공간이 필요 없으므로 chaining 방식보다 메모리 효율이 높음 ● 삽입 시 메모리 할당 오버헤드가 없으며, 메모리 할당자가 없이도 구현이 가능 ● 외부에 별도 공간을 필요로 하지 않기 때문에 chaining의 연결 리스트 같은 외부 공간에 필요한 추가적인 작업이 요구되지 않음 ● 포인터를 사용하지 않음으로써 serialization이 용이	● 효과적으로 구현하기 간단하고 기본적인 자료구조 정도만 요구됨 ● 해시 함수를 구현(선택)하는 상대적으로 쉬움. ● 테이블이 채워져도 성능 저하가 linear하게 발생 (near-filled 상태의 open-addressing 방식에서 발생하는 급작스런 lookup 시간의 증가는 발생하지 않음) ● 데이터의 크기가 대략 5 words and more 이라면, open-addressing 방식보다 적은 메모리를 사용

ITPE Information Technology Professional Engineer

[암기법]

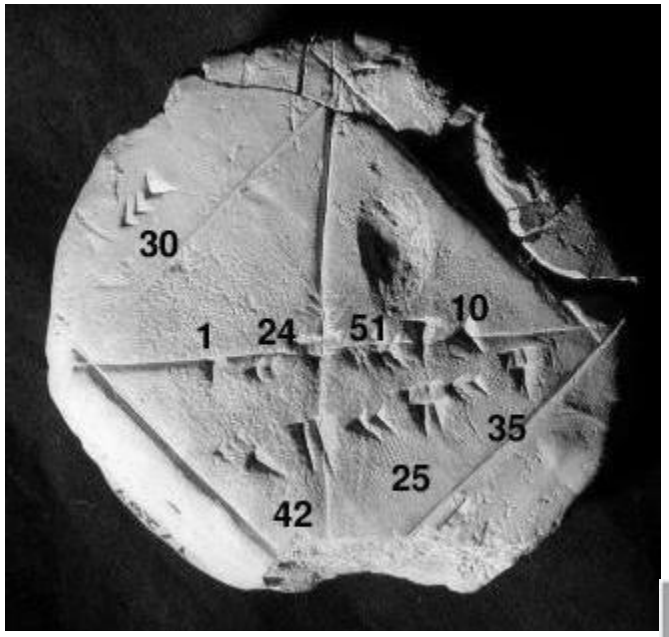
1. 해시함수 종류: 나폴리는 중(세)기이다.

2. 충돌해결방법- 오픈/클로즈 2가지 있음



ITPE Information Technology Professional Engineer

Chapter II. 압축기술



1. 압축기술
2. 동영상 압축 표준

ITPE Information Technology Professional Engineer

2. 압축기술

출제예상문제

1. 비디오 압축(Video Compression)기술을 I-프레임, P-프레임, B-프레임의 개념을 중심으로 설명하시오. (컴퓨터시스템응용, 98회, 4교시)

2. 데이터 압축기법인 런렝스(Run Length)코딩과 허프만(Huffman)코딩에 대하여 설명하시오. (컴시응, 96, 2교시)

3. 데이터를 압축할 경우의 장점과 단점에 대해 설명하시오. (정보관리, 89회 1교시)

4. 텍스트 압축기법의 유형별 장단점을 비교 설명하시오. (정보관리, 69회 1교시)

5. 영상압축기술 HEVC(High Efficiency Video Coding)의 필요성과 특징을 설명하시오. (컴퓨터시스템응용, 102회 3교시)

6. MPEG-H에 대해 Part 1,2,3 중심으로 설명하시오. (모의고사)

ITPE Information Technology Professional Engineer

I. 전송 및 저장 효율을 높이기 위한 데이터 압축의 개요

가. 데이터 압축(Data Compression)의 정의

1) 데이터의 전송 및 저장 효율을 높이기 위해 원본 데이터에서 불필요한 데이터를 삭제하거나 데이터의 중복성을 제거하여 원래의 크기보다 적은 사이즈의 데이터로 변환하는 기술

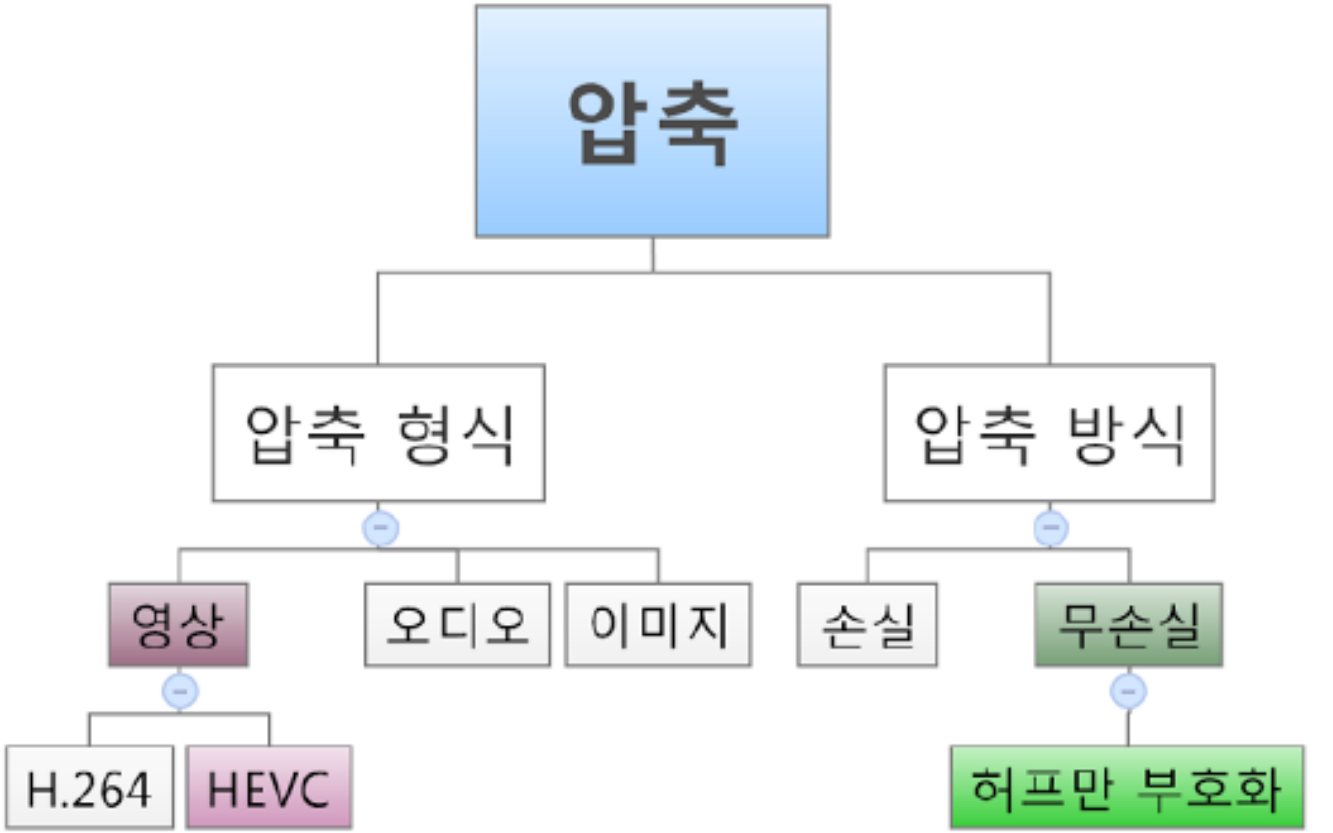
2) 데이터 압축은 원본 데이터를 그대로 복원할 수 있는 무손실 압축과 원본 데이터에 손실이 발생하는 손실 압축으로 분류

나. 데이터 압축의 특징 - 손실압축과 무손실압축의 필요성

필요이유	설명	압축율
공간절감	- 데이터 압축을 통해 저장 공간이 되는 디스크나 메모리의 점유율을 절약	- 사람의 전화음성의 경우 64KB - > 16KB - 정지컬러영상의 경우 6MB - > 1.5MB - HDTV의 경우(1.4GB - > 30MB)
속도개선	- 데이터 크기를 줄임으로 인해 복사 및 송수신 속도를 개선	
대역폭 확대효과	- 송수신 대상 데이터의 크기를 줄임으로써 단위시간당 전송량 증가로 대역폭 확대 효과를 가짐	

ITPE Information Technology Professional Engineer

II. 개념도(데이터 압축의 분류도)



ITPE Information Technology Professional Engineer

III. 데이터 압축 기술

가. 반복 순차 부호화

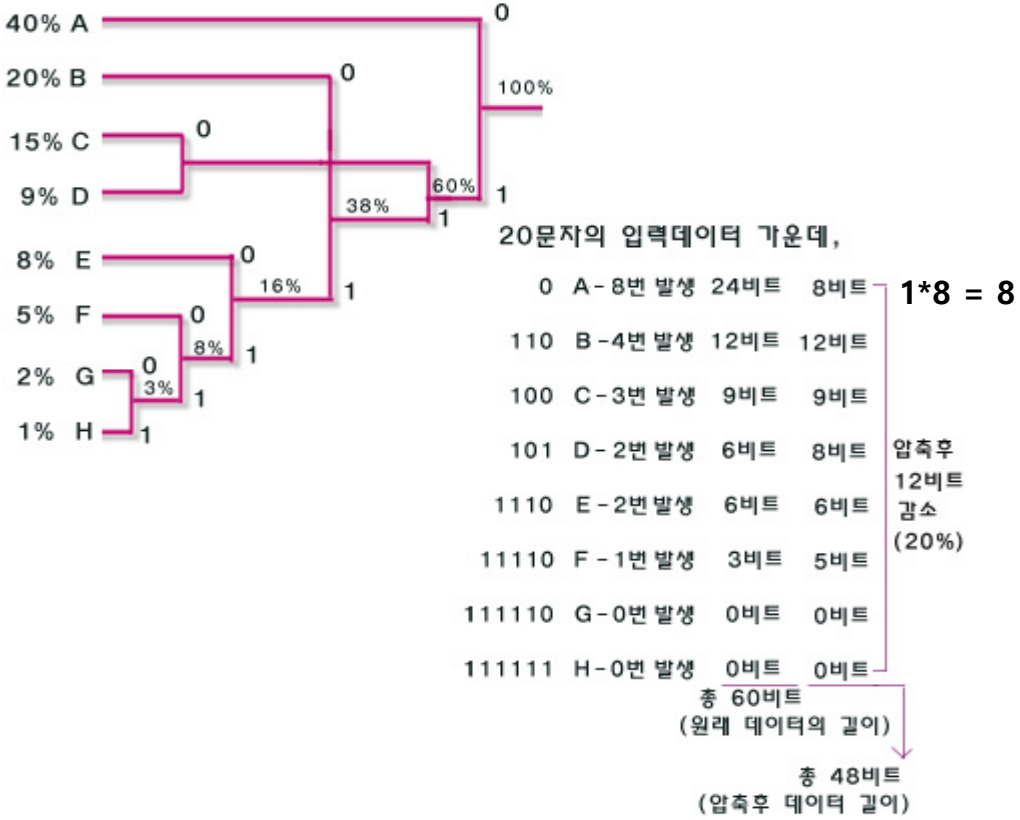
구분	내용
연속 길이 부호화 (Run-Length Coding)	- 같은 데이터가 연속적으로 나타날 경우 데이터의 값과 연속적으로 나타난 횟수로 표현하는 방법 - 팩스 데이터, 아이콘 등의 간단한 이미지와 같이 연속된 값이 많이 나타나는 데이터에 효과적 예) aaaaabbbbccccdddeeee -> a5b4c4d3e4

[참고] 이 방법은 아이콘 등의 간단한 이미지와 같이 연속된 값이 많이 있는 데이터에 효과적이다. 런 렝스 부호화는 만화나 애니메이션 등과 같이 배경의 변화가 없는 영상에 적합한 방식이다. 그러나 실제적으로 잘 적용되지 못한다. 이 방식은 3번 이상 반복되는 문자들에 대해 이용되기 때문에 영어 문장에서는 잘 수행되기 어렵다

ITPE Information Technology Professional Engineer

나. 통계적 부호화

구분	내용
허프만 부호화 (Huffman Coding)	- 가변 길이 부호화로서 자주 발생하는 데이터에는 짧은 부호를, 자주 발생하지 않는 데이터에는 긴 부호를 할당하여 전체 데이터의 길이를 원본 데이터의 길이보다 짧게 하는 방법 - 대표적인 가변 길이 부호화로써 영상 압축에 많이 사용



1.초기화 : 모든 기호를 출현 빈도수에 따라 나열한다.

2. 단 한 가지 기호가 남을 때까지 아래 단계를 반복한다.

A. 목록으로부터 가장 빈도가 낮은 것을 2개 고른다.

B. 그 다음 허프만이 두 가지 기호를 부모 노드를 가지는 부트리를 구성하고 자식노드를 생성한다.
부모 노드 단 기호들의 빈도수를 더하여 주노드에 할당하고 목록의 순서에 맞도록 목록에 삽입한다.

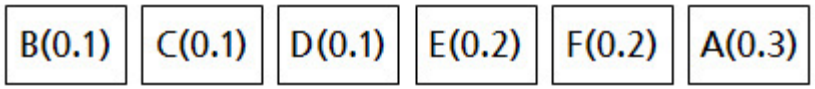
C. 목록에서 부모 노드에 포함된 기호를 제거한다.

허프만 알고리즘은 입력 기호를 잎으로 하는 이진 트리를 만들어서 접두 부호를 만들어 내는 알고리

II. 허프만 사례 – A, B, C, D, E, F 문자열

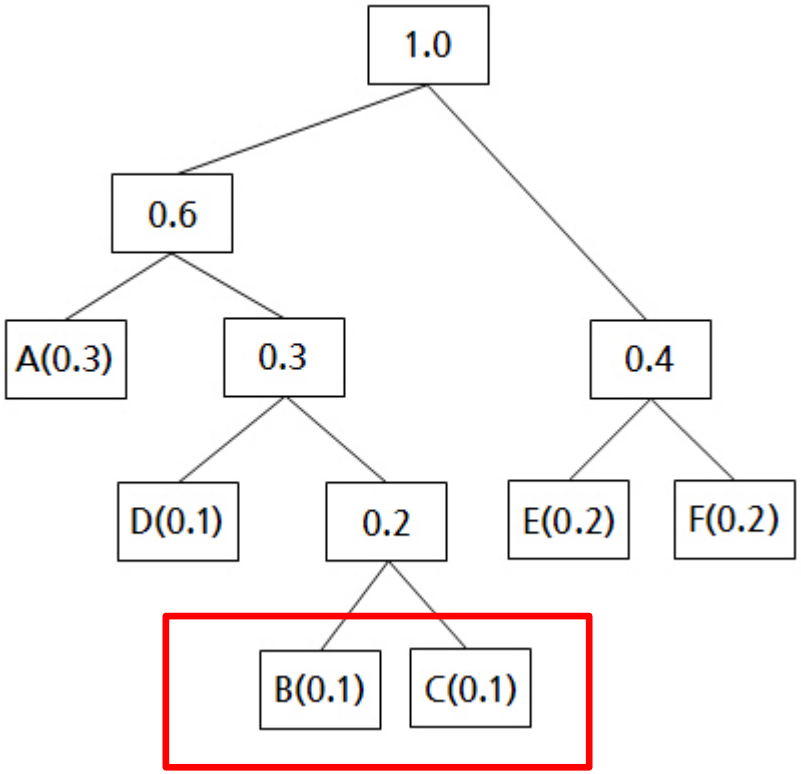
가. 초기화 : 모든 기호를 출현 빈도수에 따라 나열한다

데이터	빈도수	등장확률(가중치)
A	3	0.3
B	1	0.1
C	1	0.1
D	1	0.1
E	2	0.2
F	2	0.2

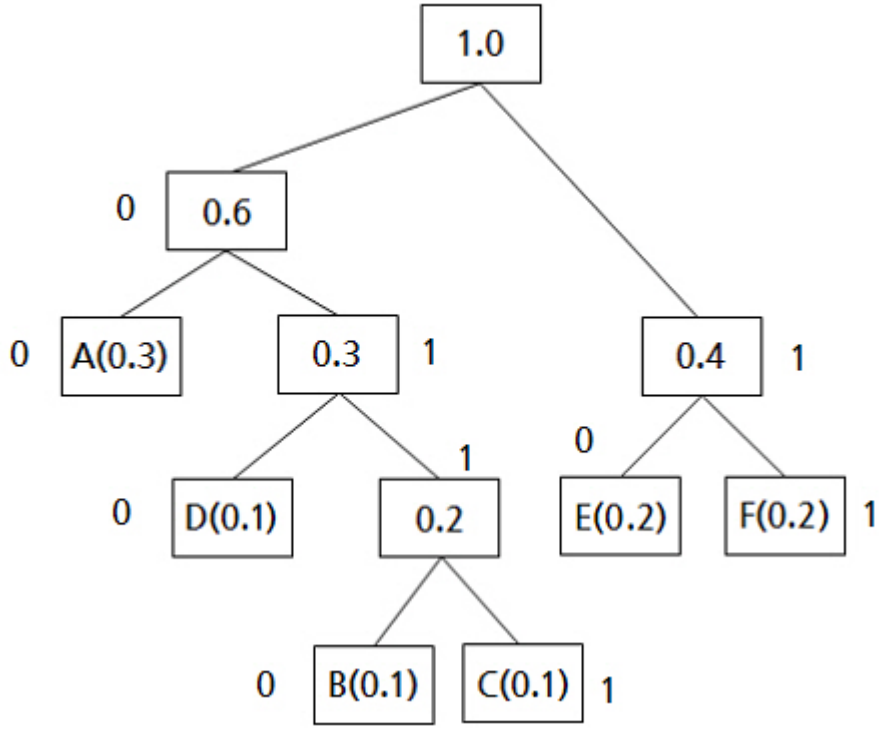


ITPE Information Technology Professional Engineer

나. 목록으로부터 가장 빈도가 낮은 것을 2개 고른다



다. 완성된 이진트리를 루트노드부터 왼쪽은 0, 오른쪽은 1을 부가해줍니다.



- 허프만 트리를 가지고 각 문자에 대한 비트를 부가합니다.
- A의 경우는 루트노드에서 왼쪽으로 한번 갔다가, 다시 한번 왼쪽으로 갔기 때문에 00 이 됩니다.
  마찬가지 원리로 B는 0110이 됩니다.

ITPE Information Technology Professional Engineer

라. 표 정리

가장 많은 빈도수의 데이터는 적은 비트, 상대적으로 적은 빈도수의 데이터들은 더 긴 비트를 서로 접두어가 겹치지 않도록 부가하여 압축하는 것입니다.

데이터	치환될 비트
A	00
B	0110
C	0111
D	010
E	10
F	11

2비트 * 3번 = 6비트 차지

(기존보다 줄이는 방법으로 접두부호(어떤 한 문자에 대한 부호가 다른 부호들의 접두어가 되지 않는 부호)를 만들어 내는 알고리즘

허프만 알고리즘은 입력 기호를 리프 노드로 하는 이진 트리를 만들어서 접두 부호를 만들어 내는 알고리즘이다.

ITPE Information Technology Professional Engineer

IV. 데이터 압축 방식

가. 정지 화상 압축 방식(image compression)

구분	설명
JPEG	- 정지 화상 압축 기술에 관한 표준 규격으로 20분의 1 정도로 압축할 수 있는 압축 방식 - 무손실 압축과 손실 압축을 모두 지원 - 화면에서 중복되는 정보를 삭제하여 정지 화상 데이터를 압축 - RBG를 허프만 코딩 테이블을 이용하여, 비트열 압축 -> 다시 복원(Decoding)
GIF	- 무손실 압축 방법을 사용 - 256 색상까지 표현 가능 - 배경을 투명하게 하거나 애니메이션 효과를 줄
PNG	- JPEG와 GIF의 장점을 조합하여 만든 형식 - JPEG의 높은 압축률과 GIF의 투명한 배경 및 (

RBG 색상표

FFFFFF	FFCCFF	FF99FF	FF66FF	FF33FF	FF00FF	66FFFF	66CCFF	6699FF	6666FF
FFFFCC	FFCCCC	FF99CC	FF66CC	FF33CC	FF00CC	66FFCC	66CCCC	6699CC	6666CC
FFFF99	FFCC99	FF9999	FF6699	FF3399	FF0099	66FF99	66CC99	669999	666699
FFFF66	FFCC66	FF9966	FF6666	FF3366	FF0066	66FF66	66CC66	669966	666666
FFFF33	FFCC33	FF9933	FF6633	FF3333	FF0033	66FF33	66CC33	669933	666633
FFFF00	FFCC00	FF9900	FF6600	FF3300	FF0000	66FF00	66CC00	669900	666600
CCFFFF	CCCCFF	CC99FF	CC66FF	CC33FF	CC00FF	33FFFF	33CCFF	3399FF	3366FF
CCFFCC	CCCCCC	CC99CC	CC66CC	CC33CC	CC00CC	33FFCC	33CCCC	3399CC	3366CC
CCFF99	CCC99	CC9999	CC6699	CC3399	CC0099	33FF99	33CC99	339999	336699
CCFF66	CCCC66	CC9966	CC6666	CC3366	CC0066	33FF66	33CC66	339966	336666
CCFF33	CCCC33	CC9933	CC6633	CC3333	CC0033	33FF33	33CC33	339933	336633
CCFF00	CCCC00	CC9900	CC6600	CC3300	CC0303	33FF00	33CC00	339900	336600
99FFFF	99CCFF	9999FF	9966FF	9933FF	9900FF	00FFFF	00CCFF	0099FF	0066FF
99FFCC	99CCCC	9999CC	9966CC	9933CC	9900CC	00FFCC	00CCCC	0099CC	0066CC
99FF99	99CC99	999999	996699	993399	990099	00FF99	00CC99	009999	006699
99FF66	99CC66	999966	996666	993366	990066	00FF66	00CC66	009966	006666
99FF33	99CC33	999933	996633	993333	990033	00FF33	00CC33	009933	006633
99FF00	99CC00	999900	996600	993300	990000	00FF00	00CC00	009900	006600

동영상 압축 방식 분류

MPEG(Moving Picture Experts Group)

동화상 전문가 그룹으로,

1988년 설립. 보통 ISO가

주로 가전 애플리케이션에 주력하는 기관으로서 동영상 압축을 위해 MPEG 표준을 정의했음

국제 전기전자통신연합

(ITU; International Telecommunication Union)

통신 어플리케이션에 주력하는 기관으로 정보를 낮은 비트율(bit rate) 화상 전화를 위한 H.24x 표준화

ITPE Information Technology Professional Engineer

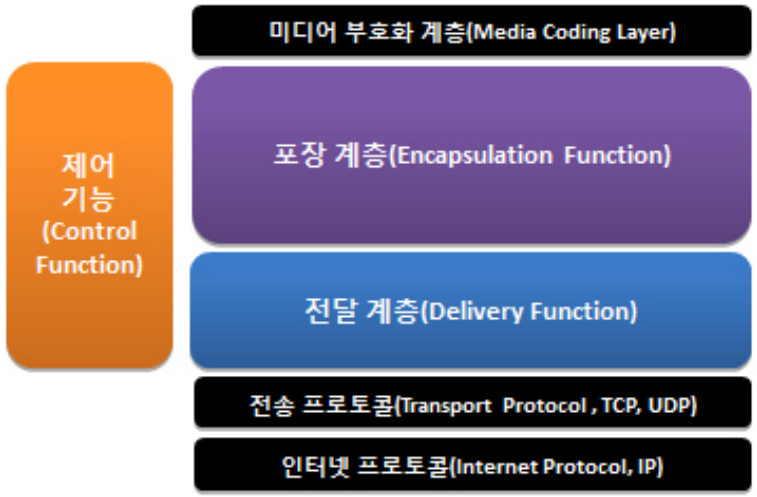
나. 동영상 압축 방식 (MPEG JTC1) : 공간적 유사성, 시간적 유사성 (Moving Picture Experts Group)

구분	설명
MPEG-1	- ISO/IEC 11172-2 - Coding of moving pictures and associated audio for digital storage media at up to about 1.5Mbit/s – Part 2:Video - 저장 미디어용 동영상 압축 부호화 - 1.5Mbps의 전송 속도로 저장 미디어인 비디오 CD 등에 적용 DCT(이산 여현 변환) 등을 사용
MPEG-2	ISO/IEC 13818-2 Generic coding of moving pictures and associated audio information: Video 범용(저장/통신/방송 미디어용) 동영상 압축 부호화 4Mbps~10Mbps에서 SDTV(표준 화질 TV), 15Mbps~30Mbps로 HDTV 화질을 실현 순행 주사 방식에 추가해서 격행 주사 방식도 수용
MPEG-4	ISO/IEC 14496-2 Coding of audiovisual objects – Part 2: Visual 멀티미디어(오디오 비주얼) 방송의 범용 동영상 압축 부호화 낮은 전송 속도에서 객체 단위의 동영상 압축 부호화를 실현 격행 주사와 휴대 전화 멀티미디어 압축에 적용 전송 속도는 10Kbps~40Mbps
MPEG-4 AVC	ISO/IEC 14496-10 Coding of audiovisual objects – Part 10: Advanced Video Coding 범용 고능률 동영상 압축 부호화 최신 부호화 기술을 집대성, H.263에 비해 2배의 압축 효율을 실현 전송 속도는 10Kbps~240Mbps

ITPE Information Technology Professional Engineer

구분	설명
MPEG-H	- MPEG-H(엠펙-에이치)의 MMT(MPEG Media Transport) - 향후 UHDTV(Ultra-High-Definition TV) 시대에 사용될 멀티미디어 부호화 표현 및 다중화 전달 표준인 MPEG-H를 표준화 중에 있다. MPEG-H는 기존 MPEG-1, MPEG-2, MPEG-4 등의 체제와 유사하게 파트 1(Part 1) 시스템, 파트 2 비디오, 파트 3 오디오 등의 체제로 구성되어 한 세트의 표준으로 발간될 예정이다. - MPEG-H의 파트 1은 MMT(MPEG Media Transport), 파트 2는 HEVC(High-Efficiency Video Coding), 파트 3는 아직 확정되진 않았으나, USAC(Universal Speech and Audio Coding) 이후 MPEG 오디오 서브그룹에서 UHDTV를 위해 표준화할 차세대 오디오 압축 부호화 표준 등이 될 것으로 예상된다. 아래는 MMT 구조입니다.

③ 제어(Control, C) 기능:
C 기능은 이러한 멀티미디어 서비스 검색 정보를 비롯한 각종 제어 정보를 전달한다



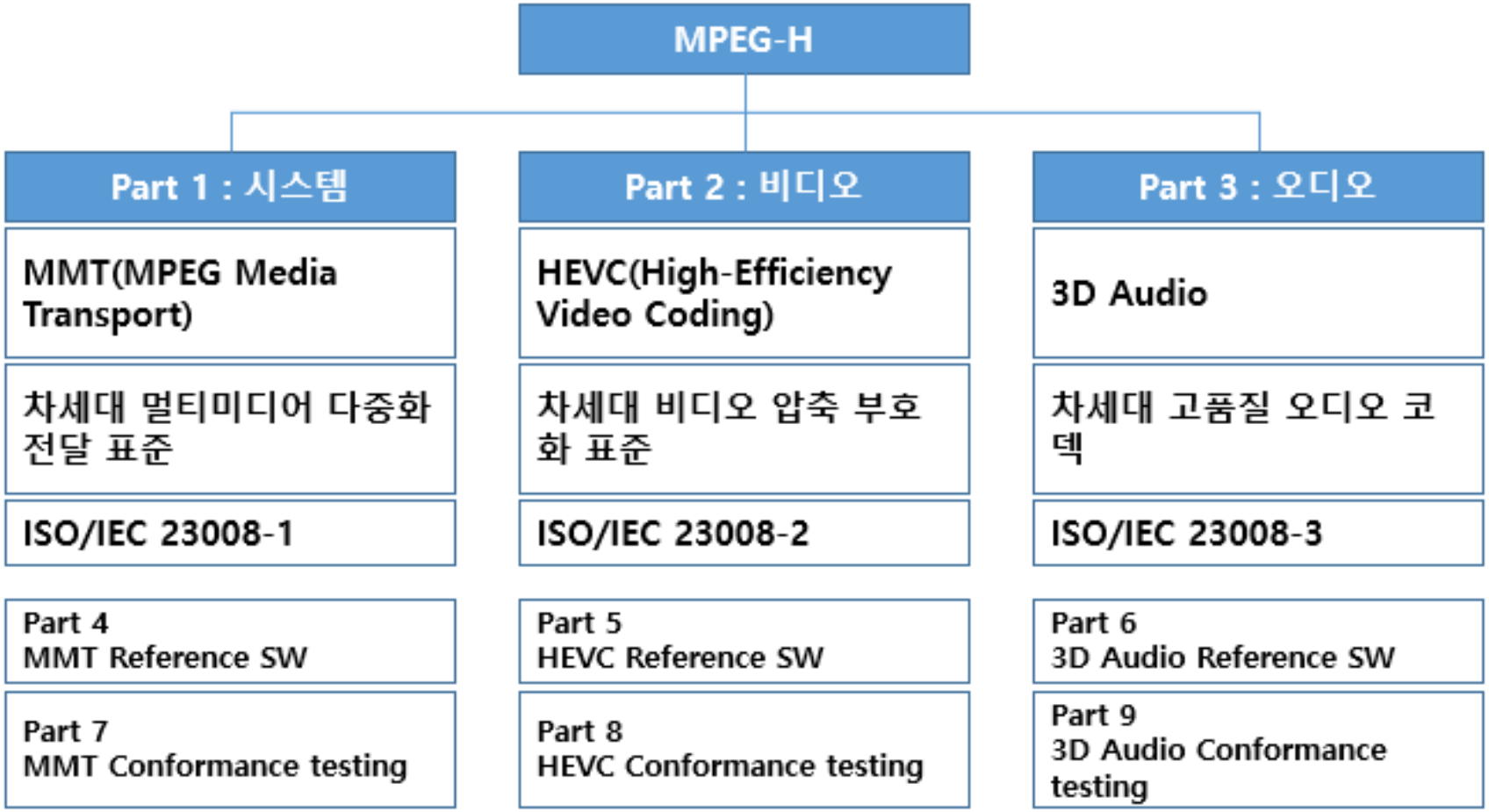
①포장(Encapsulation, E) 기능
: 멀티미디어 콘텐츠를 하나의 개체로 포장

②전달(Delivery, D): D 기능에서는 이렇게 포장된 콘텐츠를 IP 프로토콜에 맞게 패킷화하여 전송하며

CED

출처: http://www.tta.or.kr/data/weekly_view.jsp?news_id=3656

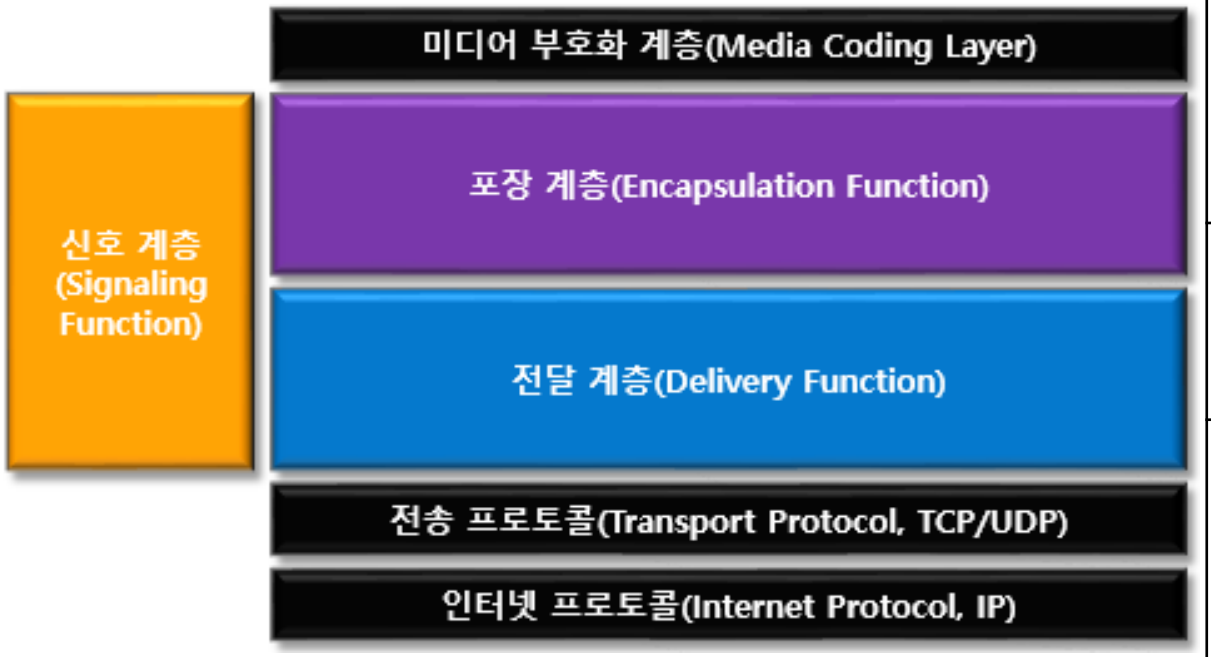
I. UHD TV 시대의 새로운 멀티미디어 전송 표준, MPEG-H의 구성



ITPE Information Technology Professional Engineer

II. MPEG-H Part 1 MMT(MPEG Media Transport)

가. MMT의 계층 구조



- E-Layer : 캡슐화(포장) 영역

E.1 / E.2 / E.3 계층으로 구성

- D-Layer : 전송 영역

D.1 / D.2 / D.3 계층으로 구성

- S-Layer : 신호 메시지 영역

S.1 / S.2 계층으로 구성

- MMT는 기존 미디어 관련 기술들을 스마트 시대에 맞게 개선시키는 목표를 가지고 표준화 진행 중임.

ITPE Information Technology Professional Engineer

II. MPEG-H Part 1 MMT(MPEG Media Transport)

나. MMT의 계층 구조 설명

Layer	기능설명	핵심기능
E-Layer	- 압축된 비디오 비트 스트림을 저장 및 전송의 효율성을 위해 캡슐화하는 기능의 Layer - 멀티미디어 컨텐츠를 하나의 개체로 포장	통합
D-Layer	- 기존의 RTP와 동일한 역할을 수행하고 안전성 및 QoS 특성에 맞는 서비스를 제공하는 Layer - 포장된 컨텐츠를 IP프로토콜에 맞게 패킷화 하여 전송	전달
S-Layer	- 미디어 소비에 대한 관리와 전송상태를 제어하고 관리하기 위한 시그널링 메시지를 정의하는 Layer - 컨텐츠 검색정보, 각종 제어정보를 전달	제어

ITPE Information Technology Professional Engineer

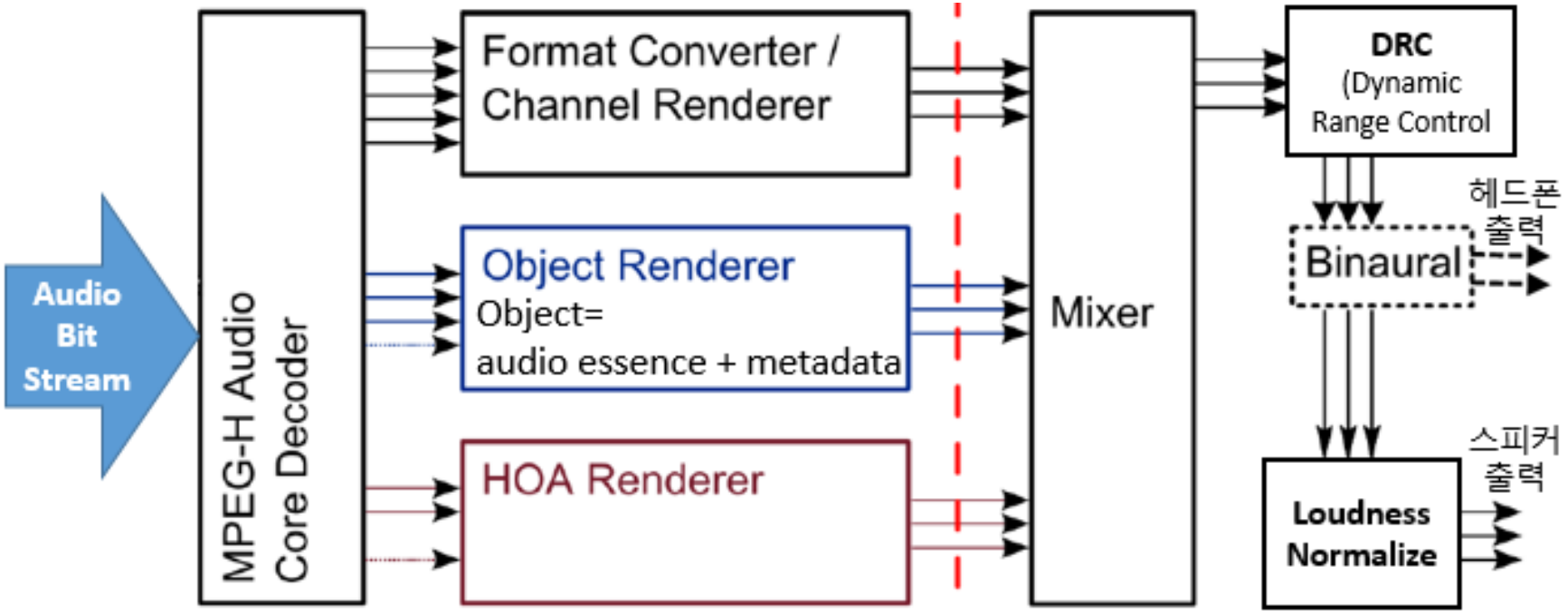
III. MPEG-H Part 2, HEVC (High Efficiency Video Coding) (HEVC의 요구 사항)

구분	요구사항	주요 특징
비트율	**기존의 비디오 표준보다 성능이 우수** 모든 비트율 범위에서 기존의 코덱보다 성능이 우수 무손실 및 시각적 무손실 기능 지원	기존 H.264/AVC 대비 2배 성능 향상 목표
해상도	QVGA(432×240)부터 최대 8kx4k까지 다양한 해상도를 지원	8K X 4K까지 지원
컬러 공간과 샘플링율	- 최대 14비트까지의 비트 심도 지원 - 넓은 컬러 gamut 영역 및 투명(transparency)채널 지원	- YCbCr 4:2:0, YCbCr 4:2:2, YCbCr/RGB 4:2:2 영상 지원
프레임율	고정 또는 0Hz부터 가변적인 프레임율 지원	- 60Hz
복잡도	사용이 예상되는 시점의 기술 수준으로 부호화/복호화 구현이 가능(복잡도는 전력 소모량/계산량/메모리량 등 포함)	- 병렬 처리를 지원, 복잡도와 부호화 효율을 조절
부호화 모드	실시간 화상 통신을 위한 저지연 모드를 지원 저장된 비디오 스트리밍 서비스를 위한 임의의 접근, 일시 정지, 고속탐색 등 기능	- 끊김 없는 실시간 영상전송 지원
스캐닝 방법	모든 프로파일과 모든 레벨에서 순차 주사 방식을 지원	- 깜박임이 적어 선명한 화질

ITPE Information Technology Professional Engineer

III. MPEG-H Part 3, 3D Audio

가. 3D Audio 아키텍처



- MPEG-H의 3D Audio Decoder Architecture. UHD 방송 표준에 대응하는 Phase1과 모바일까지 영역을 확장하는 Phase 2로 구성되며, Phase 1은 2015년 ISO/IEC 23008-3으로 국제 표준으로 정식 승인됨.

ITPE Information Technology Professional Engineer

III. MPEG-H Part 3, 3D Audio

나. 3D Audio 기술 요소

기술요소	설명	비고(기존 기술)
22.2 채널 포맷	- MPEG-H 오디오 표준화 기본 포맷 - UHDTV 사용환경을 고려한 높이 채널 지원 - 실제 재생환경에 쓸모 있는 코덱에 중점 / 채널: 스피커수	- 5.1 채널
객체 오디오 신호	- 객체: 사운드 장면을 구성하는 각 음원들(예: 기타 , 드럼 , 보컬이 3인조 밴드라면 기타, 드럼 , 보컬이 각각의 객체임) - 객체 단위로 신호를 저장하고 전송 , 사용자 단에서 다시 재구성	- 객체들이 한데 어우러진 서라운드 장면을 스피커에 1:1로 매칭
HOA(High Order Ambisonics)	- 채널과 객체 이외에 새로운 입력 신호 - 마이크를 통해 오디오 신호를 취하고, 이를 다시 재생 하는 일련의 과정에 있어서 , 특수하게 제작한 마이크와 이를 표현 하는 특수한 저장 방법을 이용	- 임의의 공간에서 Audio를 캡처하는 것이 가능
렌더링	- 객체 신호를 독립적으로 사용자 단에서 재구성하는 것 - 사용자가 22개 스피커를 가지든 , 헤드폰으로 청취하든 그 환경에 맞게 구성하는 기술	- 5.1 채널 지원을 위한 스피커와 장비 필요
Binaural 렌더링	- 모노(mono) 오디오 신호를 필터링해 헤드폰 또는 이어폰으로 재생하게 될 경우 원하는 3차원 공간상에서 재생될 때의 음장감을 느낄 수 있도록 하는 렌더링 기술	- 머리 전달 함수(HRTF)를 이용

ITPE Information Technology Professional Engineer

구분	설명
MPEG 21	**MPEG1과 MPEG2, MPEG4 등 MPEG 관련 기술을 통합하여 디지털 콘텐츠의 제작 및 유통, 보안 등의 모든 과정을 관리할 수 있게 하는 기술이다.** 콘텐츠 제작자와 유통업자, 최종 사용자가 편리하게 국제적 호환성을 가지고 콘텐츠를 식별하고 관리하며 보호할 수 있도록 하는 멀티미디어 프레임워크 핵심기술의 표준화를 목표로 한다.

다. 동영상 압축 방식 (ITU-T): 통신 업체

구분	설명
H.261	- Video codec for audiovisual services at p x 64Kbit/s - ISDN에 대응한 멀티미디어 통신(오디오 비주얼 통신)용 동영상 압축 부호화 국제 표준 - 40Kbps~2Mbps의 전송 속도로 VHS 정도의 영상 화질을 실현 - 순행 주사 방식을 채택
H.262	Information technology-Generic coding of moving pictures and associated audio information: Video ISO/IEC 표준의 MPEG-2 비디오와 명칭은 다르지만, 실제로는 같은 범용(저장/통신/방송 미디어용)의 동영상 압축 부호화 국제 표준 4Mbps~10Mbps에서 SDTV, 15Mbp~30Mbps에서 HDTV 화질을 실현 순행 주사 방식에 추가해서 격행 주사 방식도 수용
H.263	Video coding for low rate communication 공중 아날로그 전화망에서의 저속도 영상 전화/영상 회의를 위한 동영상 압축 부호화 표준 H.261의 개량본으로 약 2배의 압축 효율을 실현 전송 속도는 20Kbps~2Mbps 오류 내성(비트 오류와 패킷 손실에 강함)의 향상이 시도 됨
H.264	Advanced video coding for generic audiovisual services ISO/IEC 표준인 MPEG-4 AVC 와 명칭은 다르지만, 실제로는 같은 기술 사양의 범용 고능률 동영상 압축 부호화 표준 최신의 부호화 기술을 집대성, H.263에 비해 2배의 압축 효율을 실현 전송 속도는 10Kbps~240Mbps - IPTV 등에서 모두 사용(요새 대부분) -> HEVC로 발전 [참고] 고효율 비디오 코딩(High Efficiency Video Coding)는 H.264/MPEG-4 AVC의 성공에 힘입어 개발에 착수한 차세대 동영상 부호화 기술이다.

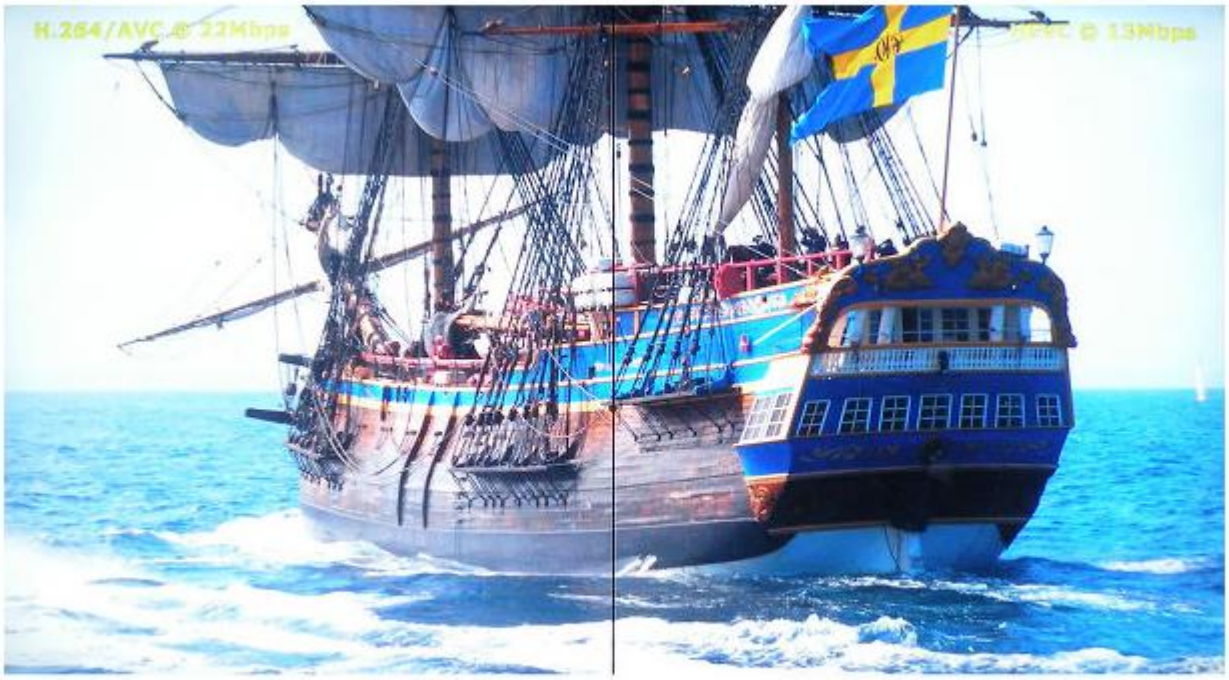
[참고] 압축 방식

v·d·e·h		
이론		엔트로피 · 복잡도 · 손실 압축 · 부호율 변형 이론 · 골룸 부호화 · 양자화 (정보 이론) · 턴스톨 부호화
무손실 압축	엔트로피 부호화	**허프만 부호화** · 산술 부호화 · 범위 부호화 · 범용 부호 · 샤논-파노 부호화 · 턴스톨 부호화
	주요 알고리즘	반복 길이 부호화 · 사전 기반 부호화 · DEFLATE · 버로우즈-휠러 변환
오디오 압축	이론	컴팬딩 · 합성곱 · 표본화 · 표본화 정리
	오디오 코덱 기술	DPCM · 푸리에 변환
	음성 코덱 기술	LPC · WLPC · CELP · ACELP · A-law · μ-law · ADPCM · MDCT
	기타	비트레이트 (가변 · 고정)
이미지 압축	용어	색 공간 · 화소 · 크로마 서브샘플링 · 해상도 · 매크로블록
	이미지 코덱 기술	프랙탈 부호화 · 웨이블릿 변환 · 이산 코사인 변환 · 카루넨-뢰브 변환 · RLE
	기타	표준 시험 영상 · PSNR · 양자화 (정보 이론)
영상 압축	용어	비디오 · 프레임 레이트 · 비월 주사 방식 · 순차 주사 방식
	영상 코덱 기술	움직임 보상 · 서브 샘플링 · 디블로킹 필터
	기타	율-왜곡 이론 · 비트레이트 (가변 · 고정)

ITPE Information Technology Professional Engineer

HEVC(High Efficiency Video Coding)

[참고] 고효율영상압축기술(HEVC: High Efficiency Video Coding)가 H.264보다 두 배 정도 압축율 뛰어남. (고효율 비디오 코딩)



3D 영상, 홀로그래피 등 고화질 영상의 최적 압축 및 전송을 위해 H.264/AVC 표준에 비해 2배 이상의 향상된 압축률을 제공하는 차세대 영상 부호화 기술(MPEG-H Part 2혹은 H.265)
2013년 1월 25일, ITU는 스위스 제네바에서 HEVC를 차세대 최종 표준안으로 승인

I. UHDTV 방송용 비디오 코덱 기술, 고효율 비디오 코딩 - HEVC 개요

정 의	특 징
- H.264/AVC 비디오 코덱 기술의 후속 기술로서 국내 지상파 UHDTV 표준에서 채택된 비디오 압축 기술	- 계층적 부호화 단위 및 구조로 부호화 - 최대 35가지 방향성을 고려한 화면내, 화면간 예측 - H.264/AVC 코덱에 비해 1/2 비트율 성능

1) 용량이 기존 H.264 방식의 인코딩을 했을 때와 비교하여, 동일 셋팅 일 경우 이론적으로 절반 용량 차지

2) HEVC 라는 이름은 ISO/IEC 표준번호 23008 part 2의 명칭으로서, MPEG측에서 사용하는 명칭

- ITU-T는 H.265라는 이름으로 2013년 4월 13일에 표준 번호 승인

ITPE Information Technology Professional Engineer

II. HEVC 주요 표준 요구사항

구 분	요 구 사 항
비트율	- 기존 비디오 표준보다 성능이 우수해야 함(약2배 향상 목표) - 모든 비트율 범위에서 기존의 코덱보다 성능이 향상 - 무손실 및 시각적 무손실 기능을 지원해야 함
해상도	- 최대 8k*4k(화소수) 까지 해상도 지원 - 가로 화소수 8,000×수직 화소 수 4,000 전후의 해상도를 지원하는 영상과 장비를 8K 라고함
컬러공간과 샘플링율	- YCbCr 4:2:2, YCbCr 4:2:2, YCbCr/RGB 4:2:2 영상 지원 - 최대 14비트까지의 비트 심도를 지원 - 넓은 컬러 gamut 영역 및 투명 채널을 지원
프레임율	- 고정 또는 0Hz부터 가변적인 프레임율을 지원
복잡도	- 사용이 예상되는 시점의 기술 수준으로 부호화/복호화 구현이 가능 (복잡도는 전력소모량, 계산량, 메모리량 등을 포함) - 병렬 처리를 지원해야함. - 복잡도와 부호화 효율을 조절할 수 있어야 함
부호화 모드	- 실시간 화상 통신을 위한 저지연 모드를 지원해야 함 - 저장된 비디오 스트리밍 서비스를 위한 임의 접근, 일시정지, 고속탐색 등의 기능
스캐닝방법	- 모든 프로파일과 모든 레벨에서 순차 주사 방식을 지원해야 함

- JCT-VC(Joint Collaborative Team on Video Coding) 는 HEVC 표준을 위해 비트율, 해상도, 컬러 공간과 샘플링율, 프레임율, 부호화 모드, 스캐닝 방법 등 총 12개 항목에 대한 요구사항 발표

- **비트율:** 통신 회선 등에서의 정보의 전송 속도를 비트 단위로 표현한 것(bps ; bit per second).
- **해상도:** 이미지를 표현하는 데 몇 개의 픽셀 또는 도트로 나타냈는지 그 정도를 나타내는 말이다.
  단위로는 1인치당 몇 개의 픽셀(pixel)로 이루어졌는지를 나타내는 ppi(pixel per inch),
  1인치당 몇 개의 점(dot)로 이루어졌는지를 나타내는 dpi(dot per inch)를 주로 사용한다.
  픽셀 또는 도트의 수가 많을수록 고해상도의 정밀한 이미지를 표현할 수 있다.
- **프레임율(frame rate):** 연속된 이미지들을 촬영하거나 재현하는 속도의 비율. 프레임 레이트는 필름의 초당 24프레임, 비디오의 초당 30프레임과 같이 초당 진행되는 프레임의 수로 측정

[참고] 연도별 비디오 기술과 표준

<table>
<tr><td rowspan="2">비디오 표준</td><td>MPEG-1(1991)
MPEG</td><td rowspan="2">H.262/MPEG-2
Part2
(1994)
MPEG & ITU-T</td><td>MPEG-4(1999)
MPEG</td><td colspan="2" rowspan="2">H.264/AVC(2003)
ITU-T & MPEG</td><td rowspan="2">HEVC
(MPEG-H
part2/H.265)
(2013.01)
ITU-T &
MPEG</td></tr>
<tr><td>H.261(1990)
ITU-T</td><td>H.263,
H.263+ (H.26L)
(1995/1996)
ITU-T</td></tr>
<tr><td>TV 기술</td><td>CableTV
(1991,95)</td><td></td><td>DigitalTV
(2001)</td><td>SatelliteTV
(2002)</td><td>IPTV
(2008)</td><td>3DTV(201x)
UDTV(201x)</td></tr>
<tr><td>저장 및 휴대기기 기술</td><td>VCD
(1993)</td><td>DVD
(1996)</td><td>Mobile VOD
(2001)</td><td>T-/S-DMB
(2004)</td><td>HD-DVD
(2006)</td><td>AT-DMB(2013)</td></tr>
</table>

- 3D TV(3-Dimensional Television): 시청각적으로 입체감을 느끼게 해 현실감과 살아 있는 듯한 느낌을 제공하는 TV
- 차세대 지상파 DMB(AT-DMB): Advanced T-DMB
- Ultra definition television

ITPE Information Technology Professional Engineer