Prim's Algorithm

```
Prim( G=(V,E) ) {
   for each vertex v in V {                                          Θ(v)
     d[v] = ∞;   p[v] = null
     inMST[v] = false   → อยู่ใน minimum spanning tree รึยัง?
   }
   select an arbitrary vertex v and let's d[v] = 0
   H = a min heap of all vertices ordered by d[]                     O(v)
  while( H ≠ ∅ ) {   O(v log v) [หมุน v ครั้ง]
     u = H.removeMin()
     inMST[u] = true
     for each v ∈ adj(u) {   ปรับ
       if( !inMST[v] AND w(u,v) < d[v] ) {
         d[v] = w(u,v);   H.decreaseKey(v)
                                O(c log
         p[v] = u
       }
     }
   }
   return p;
}
```

ใช้ binary heap

O(e log v) → dense graph → $e = \Theta(v^2)$ → $O(v^2 \log v)$

Prim's Algorithm

```
Prim( G[1..n][1..n] ) {
  for ( v = 1; v <= n; v++ ) {
    d[v] = ∞;  p[v] = null
    inMST[v] = false
  }
  select an arbitrary vertex v and let's d[v] = 0

  for ( i = 1; i <= n; i++) {
    u = minIndex( d ) ; d[u] = ∞
    inMST[u] = true
    for ( v = 1; v <= n; i++ ) {
      if( !inMST[v] AND G[u][v]< d[v] ) {
        d[v] = G[u][v];
        p[v] = u
      }
    }
  }
  return p;
}
```

$\Theta(n)$

ใช้ adjacency matrix

$\Theta(n^2)$ $\Theta(v^2)$

Kruskal's Algorithm

```
Krusal( G=(V,E) ) {
  D = a new group of disjoint sets
  for each v in V
    D.createNewSet(v)
  H = a min heap of all edges ordered by weights
  T = an empty list
  while ( T.size() < |V|-1 ) {
    (u,v) = H.removeMin()
    if ( D.findSet(u) ≠ D.findSet(v) ) {
      T.add( (u,v) )
      D.unionSet( D.findSet(u), D.findSet(v) )
    }
  }
  return T
}
```

$O(e)$

$O(e \log e)$

$O(e \log v)$

$O(e \log e)$

simple graphs : $e = O(v^2)$

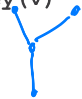
$O(e \log v)$

เหมือน prim

Dijkstra คล้าย Prim

```
Dijkstra( G=(V,E), s ) {
  for each v in V {
    d[v] = ∞;  p[v] = null
    inMST[v] = false
  }
  select an arbitrary vertex v and let's d[v] = 0
  H = a min heap of all vertices ordered by d[]
  while( H ≠ ∅ ) {
    u = H.removeMin()
    inMST[u] = true
    for each v ∈ adj(u) {  d[u]+
      if( !inMST[v] AND w(u,v) < d[v] ) {
        d[v] = w(u,v);  H.decreaseKey(v)
        p[v] = u  d[u]+
      }
    }
  }
  return p;
}
```

Dijkstra มีก็ไม่เป็นอะไร



มี negative-weight cycle ?

- relax เส้นตึงทุกเส้นเป็นจำนวน |V|-1 รอบ
- ตรวจสอบเส้นเชื่อมทุกเส้นอีกรอบ
- ถ้ายังมีเส้นตึง → มี negative-weight cycle

```
BellmanFord( G=(V,E), s ) {
  for each v ∈ V {
    d[v] = ∞; p[v] = null;
  }
  d[s] = 0            ถึง V-1
  for (i=1; i<|V|; i++)
    for each edge (u,v) ∈ E
      if ( d[u]+w(u,v) < d[v] ) {
        d[v] = d[u] + w(u,v); p[v] = u;
      }                          สิ่งอีกรอบ
  for each edge (u,v) ∈ E
    if ( d[u]+w(u,v) < d[v] ) return null
  return d;
}
```

อัลกอริทึมของ Floyd-Warshall

```
FloydWarshall( W[1..v][1..v] ) {
  D = W
  for (k = 1; k <= v; k++) {
    for (i = 1; i <= v; i++) {
      for (j = 1; j <= v; j++) {
        D[i][j] = min( D[i][j],
                       D[i][k] + D[k][j] );
      }
    }
  }
  return D;
}
```

ให้ใช้ปม {1,2,...,k} เป็นปมระหว่างทาง

$\Theta(v^3)$

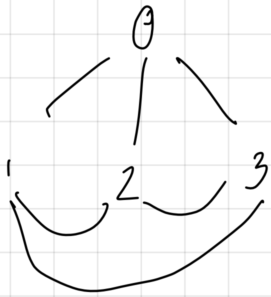
$$d_{i,j}(k) = \min(d_{i,j}(k-1),\ d_{i,k}(k-1) + d_{k,j}(k-1))$$

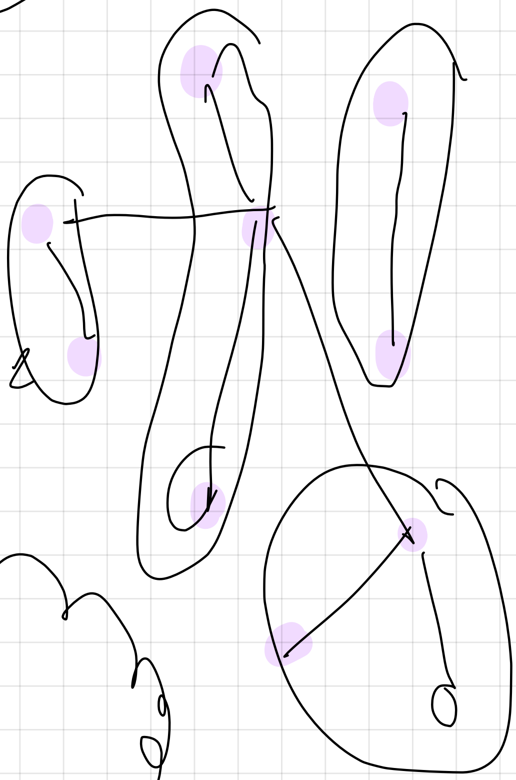
$$d_{i,j}(0) = w_{i,j}$$

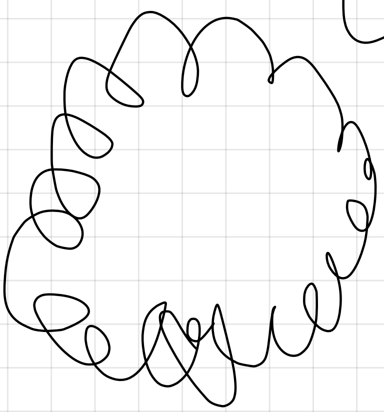
อัลกอริทึมของ Johnson

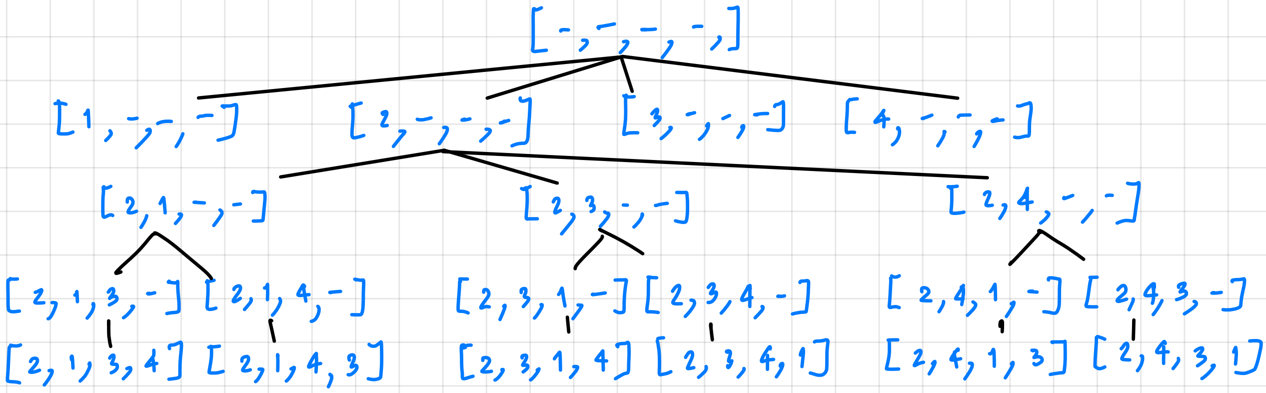
```
Johnson( G ) {
  create K : K.V = G.V ∪ {s}
           : K.E = G.E ∪ {(s,i) | i ∈ G.V)}
           : K.w = G.w, K.w(s,i) = 0, i ∈ K.V)
  h = BellmanFord(K, s)                      ← Θ(ve)
  if (h == null) return "Negative Cycle"
  else {
    for each edge (i,j) ∈ G.E
      G.w(i,j) += h[i] – h[j]
    for each vertex i ∈ G.V {
      d = Dijkstra(G, i)                     ← O(e log v)
      for each vertex j ∈ G.V
        D[i][j] = d[j] – (h[i] – h[j])
    }
    return D
  }
}
```

ถ้า Dijkstra ใช้ binary heap จะทำให้ Johnson ใช้เวลา $O(ve \log v)$ ซึ่งดีกว่า FloydWarshall เมื่อใช้กับ sparse graph


0
1
2
3







[-, -, -, -]
[1, -, -, -]
[2, -, -, -]
[3, -, -, -]
[4, -, -, -]
[2, 1, -, -]
[2, 3, -, -]
[2, 4, -, -]
[2, 1, 3, -]
[2, 1, 4, -]
[2, 3, 1, -]
[2, 3, 4, -]
[2, 4, 1, -]
[2, 4, 3, -]
[2, 1, 3, 4]
[2, 1, 4, 3]
[2, 3, 1, 4]
[2, 3, 4, 1]
[2, 4, 1, 3]
[2, 4, 3, 1]

1.1 $N_{i,j} = \min_{i \le k < j} \{ N_{i,k} + N_{k+1,j} + d_i d_{k+1} d_{j+1} \}$

A : 10×5
B : 5×30
C : 30×80
D : 80×20
E : 20×2

	1	2	3	4	5	6
1	A 0	AB = 10×5×30 = 1500	ABC min{ 0 + 12000 + (10)$\cancel{(5)}$(80) [4000] 1500 + 0 + (10)$\cancel{(30)}$(80) } [24,000] = 16000	ABCD - 0 + 20000 + (10)(5)(20) - 1500 + 48000 + (10)(30)(20) - 16000 + 0 + (10)(80)(20) = 21000	ABCDE	
2		B 0	BC = 5×30×80 = 12000	BCD min{ 0 + 48000 + (5)$\cancel{(30)}$(20), [3000] 12000 + 0 + (5)$\cancel{(80)}$(20) } [8000] = 20000	BCDE - 0 + 8000 + (5)$\cancel{(30)}$(2) [300] - 12000 + 3200 + (5)$\cancel{(80)}$(2) [800] = $\boxed{8300}$ - 20000 + 0 + (5)$\cancel{(20)}$(2) [200]	# Ans ของ 2,5
3			C 0	CD = 30×80×20 = 48000	CDE min{ 0 + 3200 + (30)$\cancel{(80)}$(2), [4800] 48000 + 0 + (30)$\cancel{(20)}$(2) } [1200] = 8000	
4				D 0	DE = 80×20×2 = 3200	
5					E 0	* แถว 6 ไม่ใช้ *
6						

1.2 295 mod 3 = 1

$d_{k+1} = d_{1+1} = d_2 = 5$

1.3 $d_2 d_2 d_6$ = ไม่มีคำตอบ ??

จาก $N_{2,5}$ คือ $d_2 d_3 d_6$

= 5 × 30 × 2 = 300

นน.	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
วัตถุ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3
2	0	0	3	4	4	7	7	7	7	7	7	7	7	7	7	7
3	0	0	3	4	15	15	18	19	19	22	22	22	22	22	22	22
4	0	0	3	4	15	15	18	19	19	22	22	22	22	22	22	25
5	0	0	3	4	15	15	18	19	19	22	22	22	23	24	35	35
6	0	0	3	4	15	15	18	19	19	22	22	22	27	27	35	35
7	0	0	3	4	15	15	18	19	19	22	22	26	27	29	35	35

2.1 : 18

2.2 : 35

6.1 n = 5

```
for i := 1 to 5
  for j := 1 to 5
    for k := 1 to 5
      print "OK"
```

ดู loop ทุก loop ทำ 1,...,n รวม n รอบ

$\Theta(125)$: $\boxed{\Theta(n^3)}$ ✗✗

ผลการรัน

OK
OK
OK
OK
⋮
OK

$\}$ 125 ตัว

6.2

```
int A(5)
  if (n == 0)
    return 1;
  else
    return A(n/2);
}
```

$\boxed{\Theta(\log_2 n)}$

ผลการรัน

A(5)
return ↓
A(2)
return ↓
A(1)
return ↓
A(0) return 1

input 5

$\}$ 4

6.3 A(5)

for 0,..,4

0 A(4)
1 A(4)
2 A(4)
3 A(4)
4 A(4)

A(4) ← A(3) $\}$ 4 รอบ

A(3) ← A(2) 3 รอบ

A(2) ← A(1) 2 รอบ

A(1) 1 รอบ return 1

$\boxed{\Theta(n!)}$

5 × 4 × 3 × 2 × 1

6.5

$f(1) = 295 \bmod 4 = 3$

$f(2) = 295 \bmod 3 = 1$

$f(3) = f(2) + 2 * f(1) + 2 = 1 + 2(3) + 2 = 9$

$f(4) = f(3) + 2 * f(2) + 2 = 9 + 2(1) + 2 = 13$

$f(5) = f(4) + 2 * f(3) + 2 = 13 + 2(9) + 2 = 33$

$f(6) = f(5) + 2 * f(4) + 2 = 33 + 2(13) + 2 = 61$

$f(7) = f(6) + 2 * f(5) + 2 = 61 + 2(33) + 2 = 129$

$f(8) = f(7) + 2 * f(6) + 2 = 129 + 2(61) + 2 = 253$ // Ans

6.6.1)

$T1 = 1$

$T2 = 2$

$T3 = n-1$

$T4 = n-1$

$T5 = 1$

$T_6 = 1$

6.6.2) ยังไม่แน่ใจ

6.6.3 $1 \times 2 \times (n-1)^2 \times 1 \times 1 = \Theta(n^2)$

6.7.1) $\log\log n < \log n < n^{1.15} < n\log n < n^2$

6.7.2) $0.5^n < 2.5 < 2^{1.5} < \log n < n < n\log n$

$2^{1.5} \sim 2.8$

6.8

```
for (i=1; i<n; i++)        → Θ(n-1) * 1,...,n-1
  for (j=1; j<n; j++)      → Θ(n-1) * 1,...,n-1
    sum++                  → Θ(1)
```

$\Theta(n)$, $\Theta(n^2)$

```
for (i=1; i<n; i++)        → Θ(n-1)
  for (j=1; j<i; j++)      → Θ
    sum++                  → Θ(1)
```

i=1 2 3 4 ... n-1

1 1 1 ... 1

2 2 2

3 :

$i-1 = n-2$

$\Theta(n^2)$

$\therefore \Theta(n^2)$

6.9

```
for (i=1; i<n; i=i+2)      → Θ(n/2)
  for (j=1; j<n; j=j+2)    → Θ(n/2)
    sum++                  → Θ(1)
```

$\Theta(n^2)$

```
for (i=1; i<n; i=i+2)
  for (j=1; j<i; j=j+2)    → j<i
    sum++
```

loop นี้ น้อยกว่า loop แรก

$\therefore \Theta(n^2)$

$$j = (2^1 - 1) + (2^2 - 1) + (2^3 - 1) + \dots (2^{\log_2 n} - 1) = (2^1 + 2^2 + \dots + 2^{\log_2 n}) - \log_2 n$$

$$i = \quad 1 \quad 2 \quad 4 \quad 8 \quad \boxed{n}$$

$$1 \quad 2 \quad 4 \quad \dots \quad n$$

$$= 2^{\log_2 n} - \log_2 n$$

$$=$$