

アルゴリズムとデータ構造

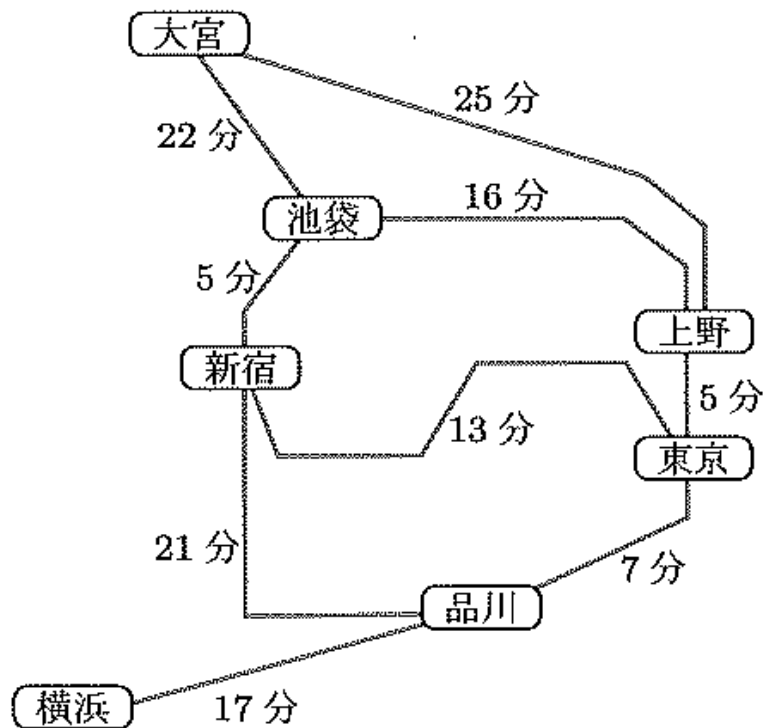
第19週目

担当 情報システム部門 徳光政弘
2025年10月28日

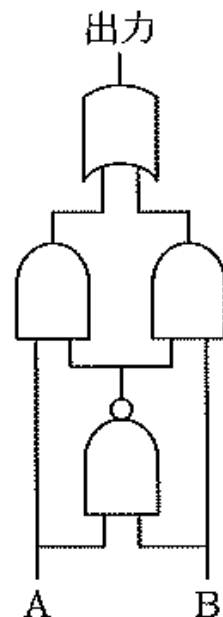
今日の内容

- グラフのデータ表現
- ダイクストラ法
 - ある頂点からの最短経路を求める

グラフ



(a) 鉄道の路線図



(b) 回路図

図 10.1 一般生活におけるグラフ

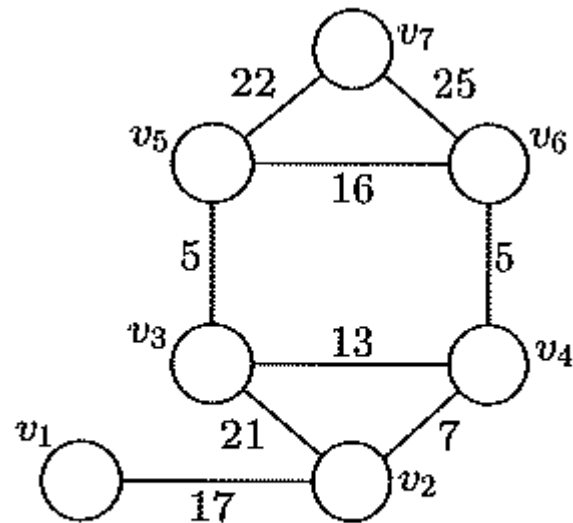
最短経路問題

例えば、 v_1 から v_7 までの最短となるコスト(重み)を求めたい。

最小のコストと経路

経路

コスト 54 $(v_1, v_2, v_4, v_6, v_7)$



(a) 路線図を表すグラフ

最短経路問題

乗り換え案内

乗車駅	<input type="text" value="横浜"/>
下車駅	<input type="text" value="大宮"/>

検索

最短経路の
検索を実行

最短の乗り換え方法は以下のとおりです



横浜
 ○○線 24 分 ×× 円
東京
 ○○線快速 30 分 ×× 円
大宮

図 10.7 最短経路問題の実用例

最短経路問題(到着時刻順)



最短経路問題(乗換回数順)

到着時刻順

乗換回数順

料金の安い順

↓ルート1	08:29→14:06 5時間37分	21,710円 乗換: 1回	楽
↓ルート2	08:29→14:33 6時間4分	21,570円 乗換: 1回	楽
↓ルート3	09:35→15:15 5時間40分	21,710円 乗換: 1回	楽

6件中1~3件を表示しています。 [次の3件 >](#)

ルート1

08:29発→14:06着 5時間37分 (乗車5時間28分)

乗換: 1回

現金優先: 21,710円 (乗車券11,880円 特別料金9,830円)

892km

+ ルート保存

定期券

ルート共有

印刷する

08:29 発 米子 時刻表 地図

6駅

J R 特急やくも8号
岡山行
[発] 情報なし → [着] 3番線

11,880円
指定席: 2,730円

10:43着
10:52発

岡山 時刻表 地図

7駅

J R 新幹線のぞみ128号
東京行
[発] 23番線 → [着] 17番線

指定席: 7,100円

14:06 着 東京 時刻表 出口 地図

PR

7

ダイクストラ法

- ① 集合 S に含まれない頂点のうち、始点からの距離を表す変数 d_i の値がもっとも小さい頂点 v_k を選ぶ.
- ② 頂点 v_k を S に加える.
- ③ 頂点 v_k に隣接する頂点のうち S に含まれないすべての頂点について、始点から距離を格納する変数 d_i を再計算する.

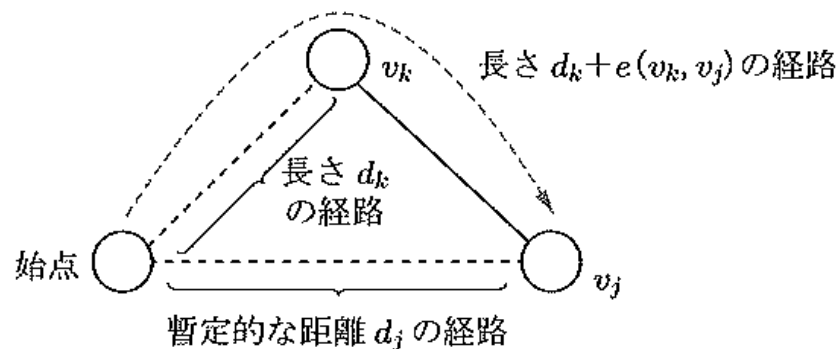


図 10.8 変数 d_j の再計算

ダイクストラ法

初期値: $S = \phi$,

d_1	d_2	d_3	d_4	d_5	d_6	d_7
0	∞	∞	∞	∞	∞	∞

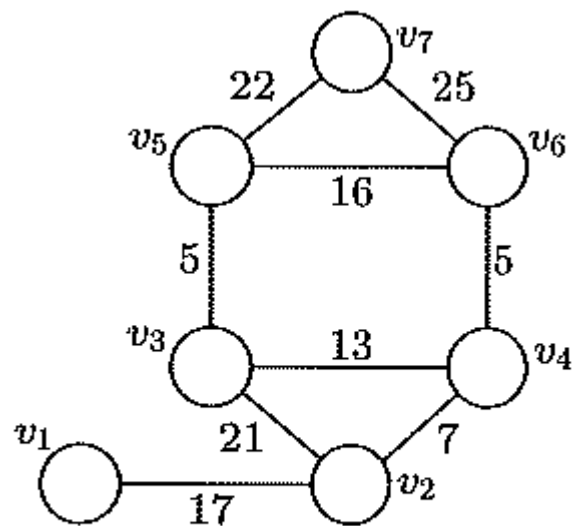
1回目: $S = \{v_1\}$,

d_1	d_2	d_3	d_4	d_5	d_6	d_7
0	17	∞	∞	∞	∞	∞

2回目: $S = \{v_1, v_2\}$,

d_1	d_2	d_3	d_4	d_5	d_6	d_7
0	17	38	24	∞	∞	∞

$$d_3 = \min\{\infty, 17 + 21\} = 38, \quad d_4 = \min\{\infty, 17 + 7\} = 24$$



(a) 路線図を表すグラフ

ダイクストラ法

3回目: $S = \{v_1, v_2, v_4\}$,

d_1	d_2	d_3	d_4	d_5	d_6	d_7
0	17	37	24	∞	29	∞

4回目: $S = \{v_1, v_2, v_4, v_6\}$,

d_1	d_2	d_3	d_4	d_5	d_6	d_7
0	17	37	24	45	29	54

5回目: $S = \{v_1, v_2, v_3, v_4, v_6\}$,

d_1	d_2	d_3	d_4	d_5	d_6	d_7
0	17	37	24	42	29	54

6回目: $S = \{v_1, v_2, v_3, v_4, v_5, v_6\}$,

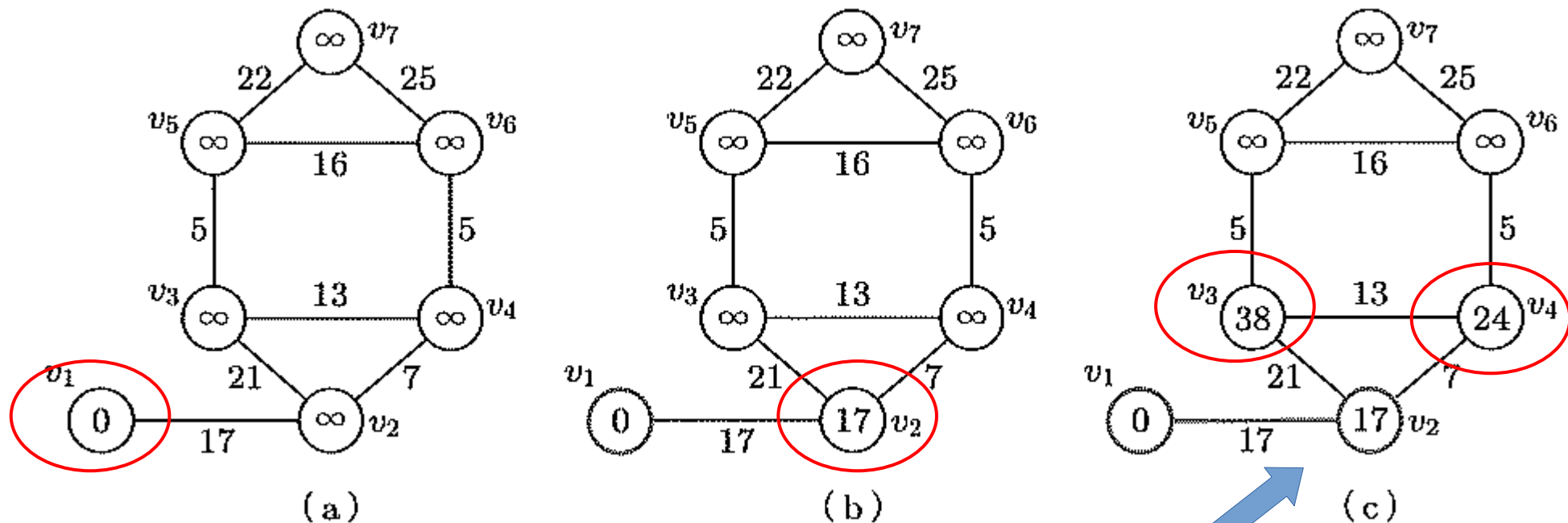
d_1	d_2	d_3	d_4	d_5	d_6	d_7
0	17	37	24	42	29	54

7回目: $S = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$,

d_1	d_2	d_3	d_4	d_5	d_6	d_7
0	17	37	24	42	29	54

ダイクストラ法

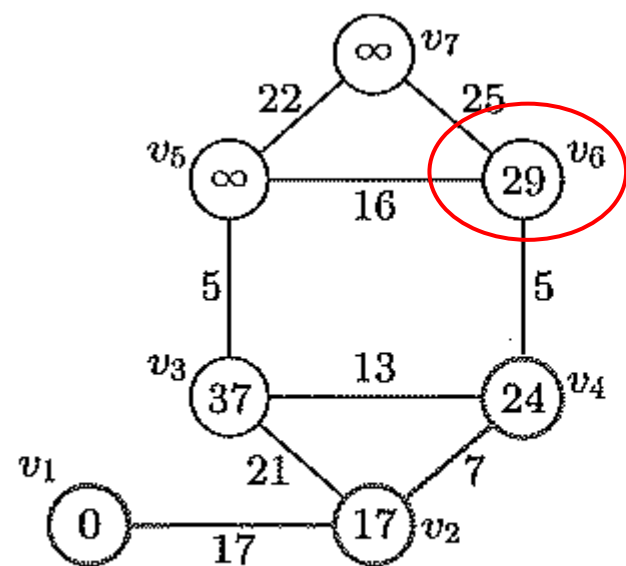
図10.9は色付きでわかりやすい



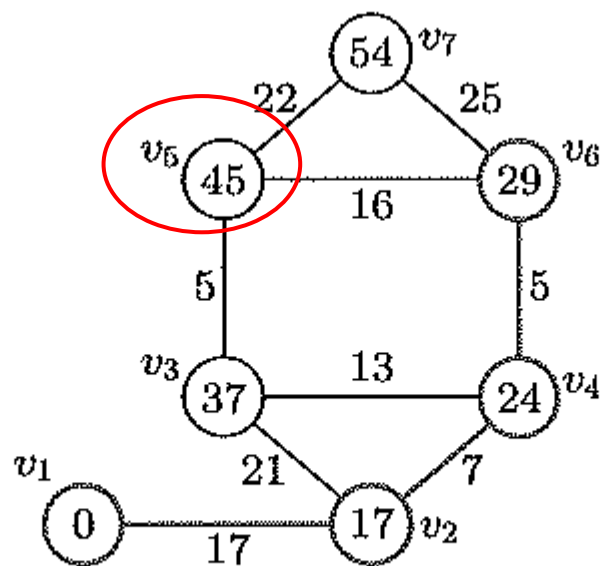
$$d_3 = \min\{\infty, 17 + 21\} = 38, \quad d_4 = \min\{\infty, 17 + 7\} = 24$$

最小コストを比較している演算になっている。

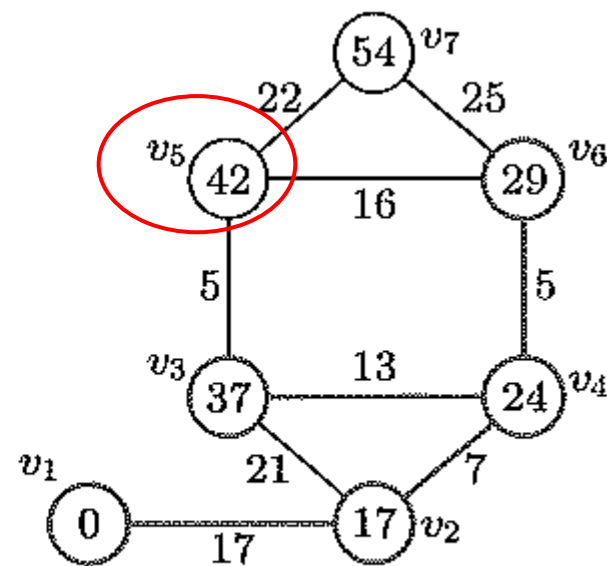
ダイクストラ法



(e)

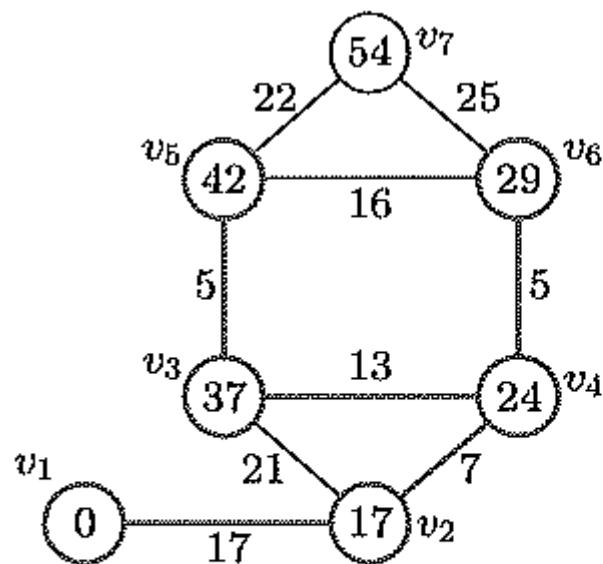


(f)

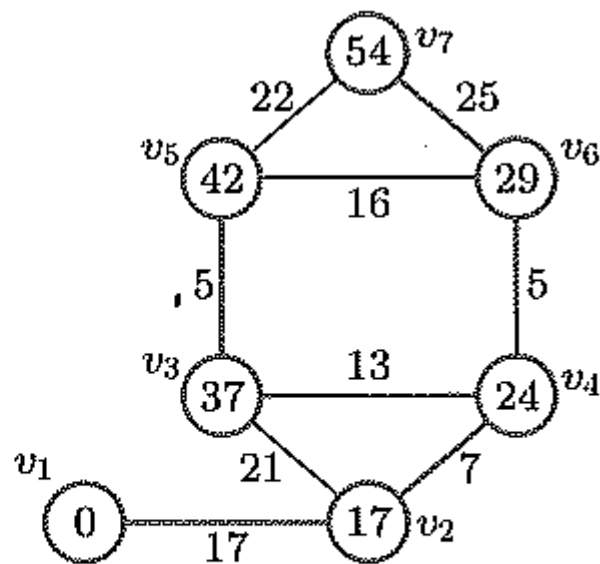


(g)

ダイクストラ法



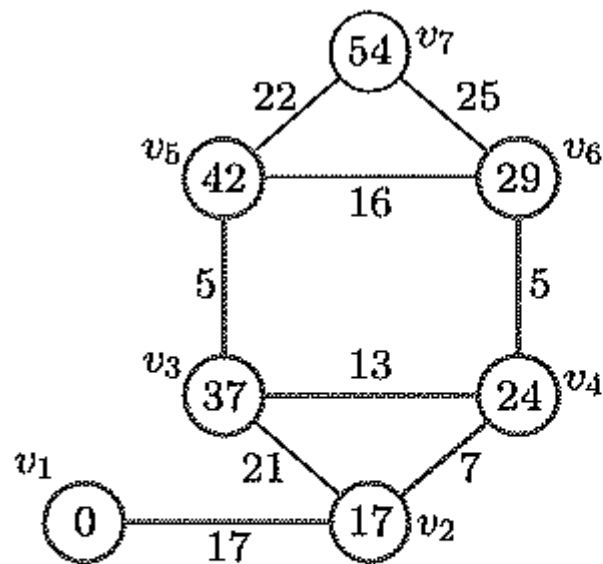
(h)



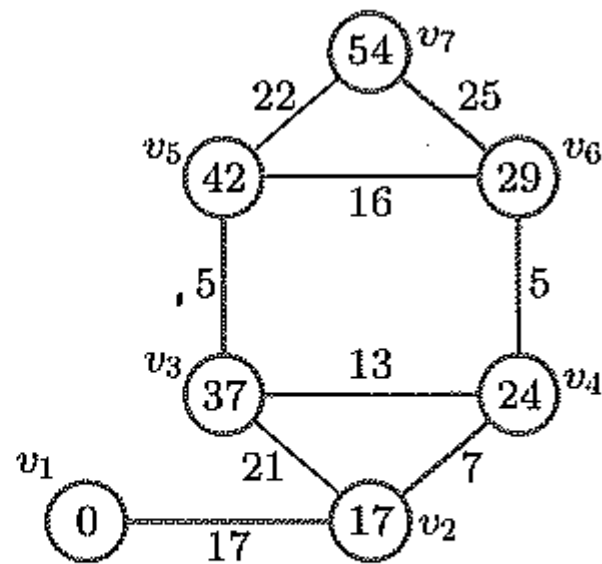
(i)

図 10.9 ダイクストラ法の実行例

ダイクストラ法



(h)



(i)

図 10.9 ダイクストラ法の実行例

頂点集合Sに対する計算のみを実行している

ダイクストラ法

アルゴリズム 10.3 ダイクストラ法

入力：頂点の集合 V と辺の集合 E ，および V に含まれる始点 v_s 。

(各頂点 v_i は $v[i]$ と表し，辺 (v_i, v_j) の重みは $e[i][j]$ で表す.)

```
for (i=1; i<=n; i=i+1) { D[i]=\infa; }
```

```
S= $\phi$ ; s=1; D[s]=0;
```

```
for (i=1; i<=n; i=i+1) {
```

```
    S に含まれない頂点の中から，配列 D の値が最小の頂点  $v[k]$  を求める;
```

```
    頂点  $v[k]$  を S に追加する;
```

```
    for (j=1; j<=n; j=j+1) {
```

```
        if (( $v[j]$  が  $v[k]$  に隣接する) かつ ( $v[j]$  が S に含まれない))
```

```
            { D[j]=min(D[j], D[k]+e[k][j]); }
```

```
    }
```

```
}
```

実装では、訪問のチェックに配列、隣接行列を使うのがやりやすい。

ダイクストラ法

- for文が2個ある 少なくとも $O(n^2)$
- 頂点に対するコストの計算(Sと配列) $O(n)$
- 合計すると、多項式の次数が優先されて $O(n^2)$