國立高雄第一科技大学
National Kaohsiung First University of Science and Technology

Efficient Graph-Based Image Segmentation

Pedro. F. Felzenszwalb, and Daniel P. Hutenlocher

Intl. Journal of Computer Vision (IJCV), 2004

Speaker: Shih-Shinh Huang

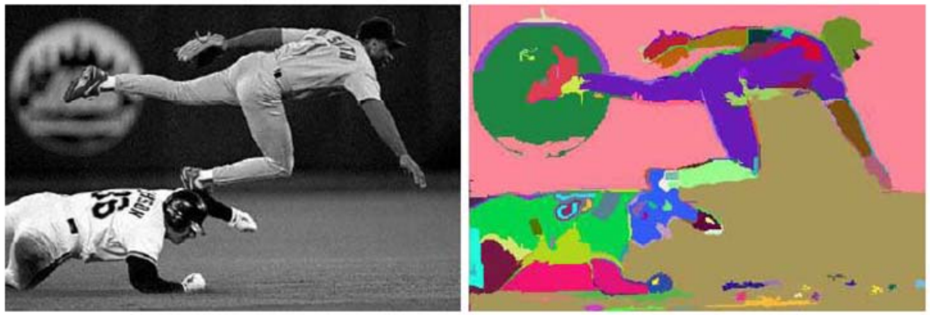
August 17, 2018

Outline

- Introduction
- Graph-Based Formulation
- Partition Strategy
- Segmentation Algorithm

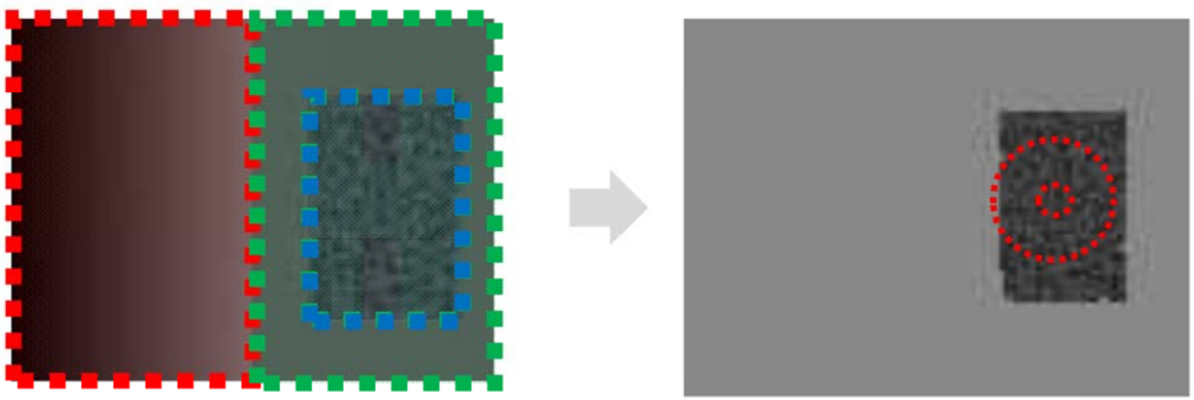
Introduction

- About Segmentation
  - partition an image into a set of disjoint regions.
  - facilitate the process of addressing a wide range of vision problems.
    - Intermediate-Level: motion estimation
    - High-Level: image indexing or object detection



Introduction

- Observation
  - It is not adequate to assume that regions have nearly constant or slowly varying intensities.
  - The determination of boundary between regions cannot only use local decision criteria.



region with high variability

Introduction

- Objective
  - develop image segmentation approach that
    - captures perceptually important regions that reflect global aspect.
    - runs efficiently in time nearly linear in the number of image pixels.

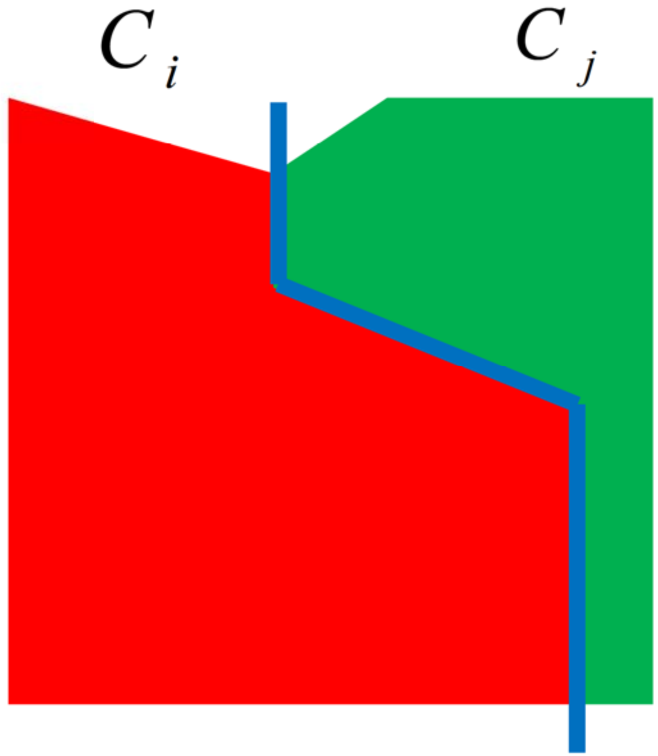
Introduction

- Idea: adaptive criteria
  - There is a boundary between two adjacent regions C_i and C_j

    $\text{Between}(C_i, C_j) > \text{Within}(C_i)$ or

    $\text{Between}(C_i, C_j) > \text{Within}(C_j)$

    - Between: difference across two regions
    - Within: difference (or variation) within a region



Graph-Based Formulation

- Graph Representation
  - Let $G = (V, E)$ be an undirected graph
    - $v_i \in V$: set of vertices (pixels) to be segmented
    - $e = (v_i, v_j) \in E$: set of edges corresponding to pairs of neighboring vertices (pixels)
    - Each edge e $= (v_i, v_j) \in E$ has a weight $w(v_i, v_j)$ denoting the dissimilarity between v_i and v_j

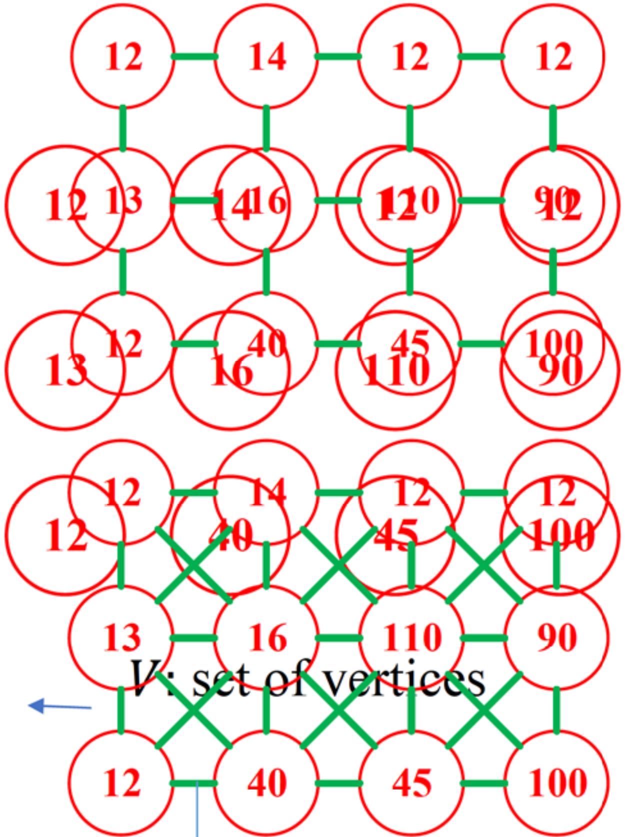
Graph-Based Formulation

- Graph Representation

12	14	12	12
13	16	110	90
12	40	45	100

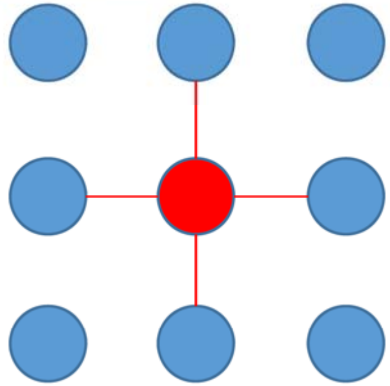
$w(.) = |12 - 13| = 1$

$w(.) = |12 - 40| = 28$

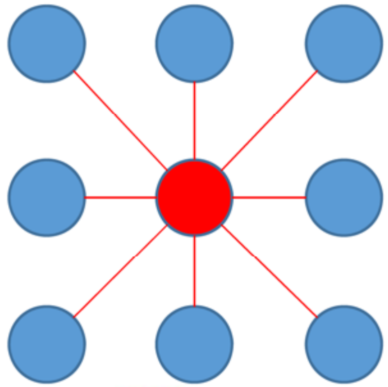


V: set of vertices

E: set of edges



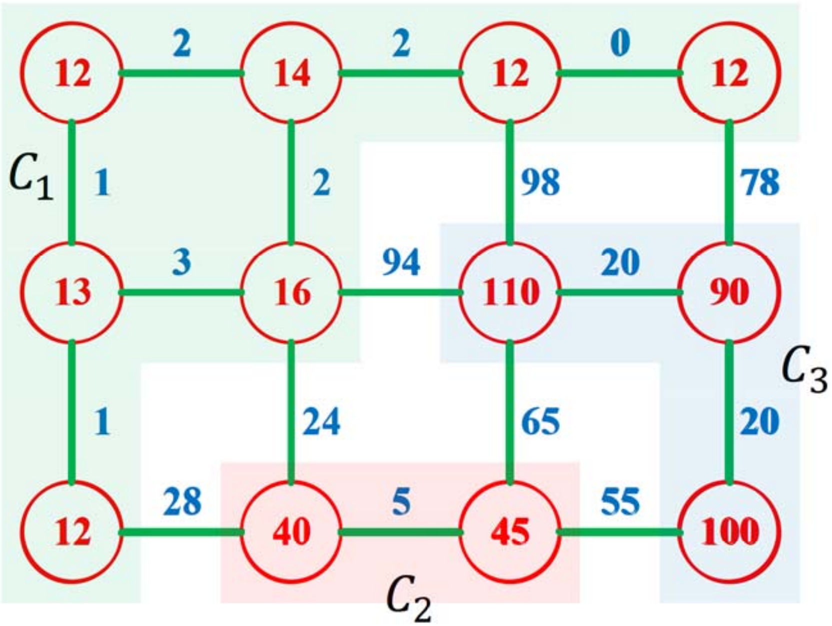
N_4 neighborhood



N_8 neighborhood

Graph-Based Formulation

- Segmentation Formulation
  - partition the vertex set V of a graph into components $C_1, C_2, \ldots\ldots$



- Edges between two vertices in the same component should have lower weights
- Edges between vertices across different components should have higher weights

Partition Strategy

- Internal (Within) Difference $Int(.)$
  - Definition: largest weight in the minimum spanning tree (MST) of a component $C \subseteq V$

$$Int(C) = \max_{e \in MST(C,E)} w(e)$$

  - $Int(C) = 0$ if C has only one pixel

Partition Strategy

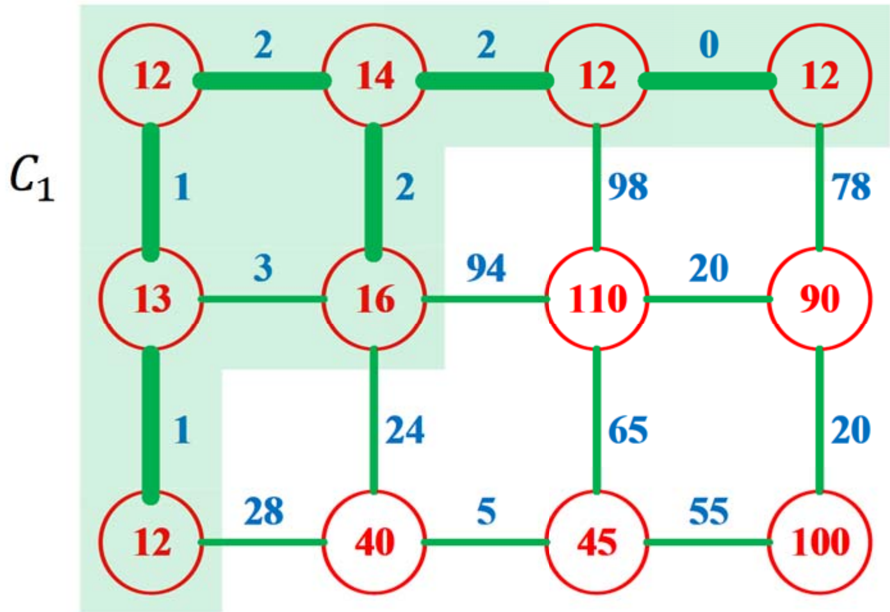
- Component (Between) Difference $Dif(.)$
  - Definition: minimum weight of edges connecting two components $C_i \subseteq V, C_j \subseteq V$

$$Dif(C_j, C_j) = \min_{v_i \in C_i, v_j \in C_j, (v_i, v_j) \in E} w(v_i, v_j)$$

  - $Dif(C_i, C_j) = \infty$ if there is no edge connecting C_i and C_j

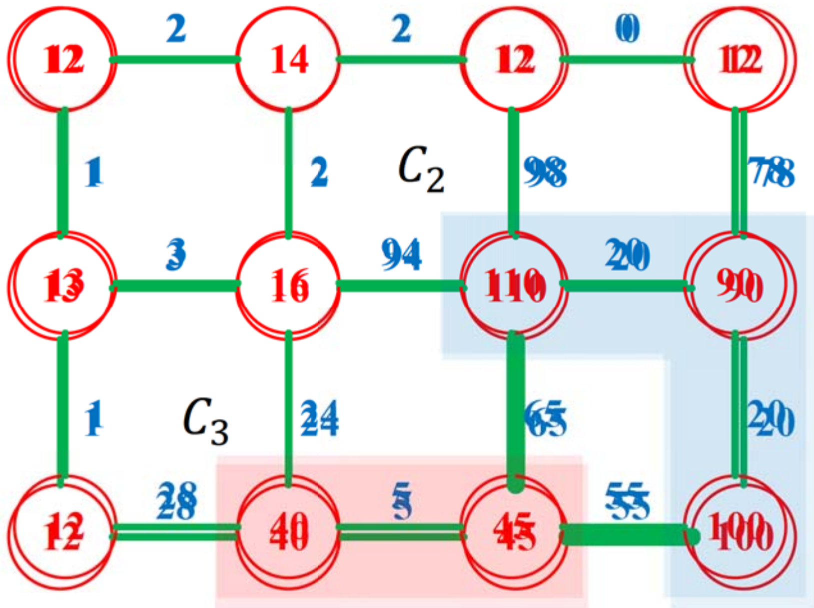
Partition Strategy

Internal Difference



$Int(C_1) = \max_{e \in MST(C,E)} w(e) = \max\{2,2,0,1,2,1\} = 2$

Component Difference



$Dif(C_2, C_3) = \min_{v_i \in C_2, v_j \in C_3, (v_i, v_j) \in E} = \min\{65, 55\} = 55$

Partition Strategy

- Boundary Predicate
  - evaluate if there is evidence for a boundary between a pair of adjacent components.

$$D(C_i, C_j) = \begin{cases} true & Dif(C_i, C_j) > Int(C_i) \text{ or } Dif(C_i, C_j) > Int(C_j) \\ false & \text{otherwise} \end{cases}$$

$$D(C_i, C_j) = \begin{cases} true & Dif(C_i, C_j) > \min\{Int(C_i), Int(C_j)\} \\ false & \text{otherwise} \end{cases}$$

Partition Strategy

- Boundary Predicate
  - This predicate is not a good estimate of local property
    - makes the algorithm tend to have components with small size.
    - Extreme Case: $Int(C) = 0$ if $|C| = 1$

Partition Strategy

- Boundary Predicate
  - add a threshold function $\tau(.)$ based on component size, that is, $\tau(C) = \frac{k}{|C|}$

Rule: $Dif(C_i, C_j) > \min\{Int(\mathrm{C_i}), Int(\mathrm{C_j})\}$

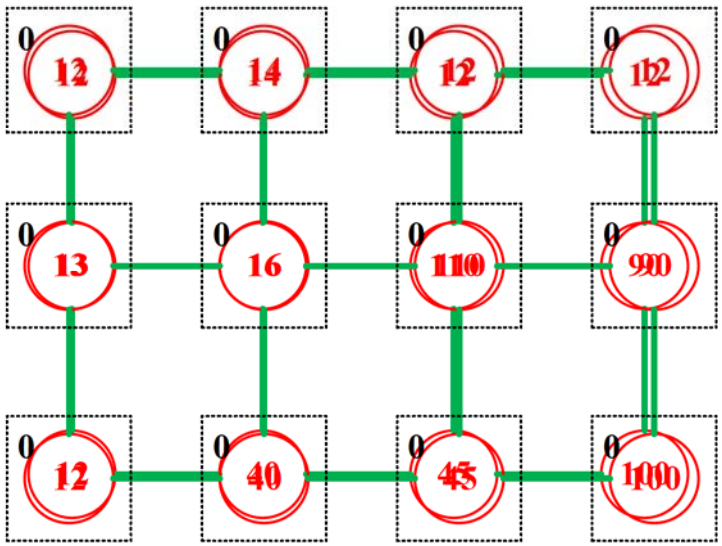
$$\Rightarrow \quad Dif(C_i, C_j) > \min\{Int(C_i) + \frac{k}{|C_i|}, Int(\mathrm{C_j}) + \frac{k}{|C_j|}\}$$

Segmentation Algorithm

- General Description
  - Input:
    - a graph $G = (V, E)$ with n vertices and m edges
    - a constant parameter k
  - Output:
    - a partition of V into components $S = (C_1, C_2, .., C_r)$

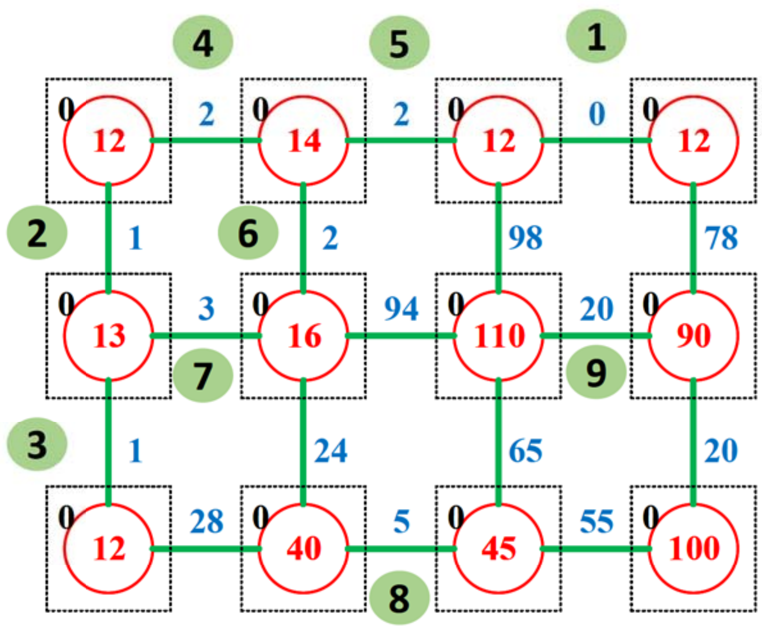
Segmentation Algorithm

- Initialization
  - consider each vertex as a single-element component $S = (C_1, C_2, \dots C_n)$
  - initialize each component with $Int(C_i) = 0$



Segmentation Algorithm

- Initialization
  - sort all edges e ∈ E into $(e_1, e_2, .. e_m)$ according to their weights in a non-decreasing order



Segmentation Algorithm

- Iteration Step ($q = 1,2,..,m$)
  - Step 1: take the edge $e_q = (v_i, v_j)$, where $v_i \in C_i$ and $v_j \in C_j$
  - Step 2: if $C_i \neq C_j$
    - Step 2.1: if boundary predicate $D(C_i, C_j)$=false. merge the components C_i and C_j
    - Step 2.2: if C_i and C_j are merged, set $Int(C_i \cup C_j)$=$w(e_q)$
  - Step 3: $q \leftarrow q + 1$ and go to Step 1

Segmentation Algorithm

- Iteration Step (Merge Condition)

$$D(C_i, C_j) = false \text{ if } Dif(C_i, C_j) \leq \min\{Int(C_i) + \frac{k}{|C_i|}, Int(C_j) + \frac{k}{|C_j|}\}$$

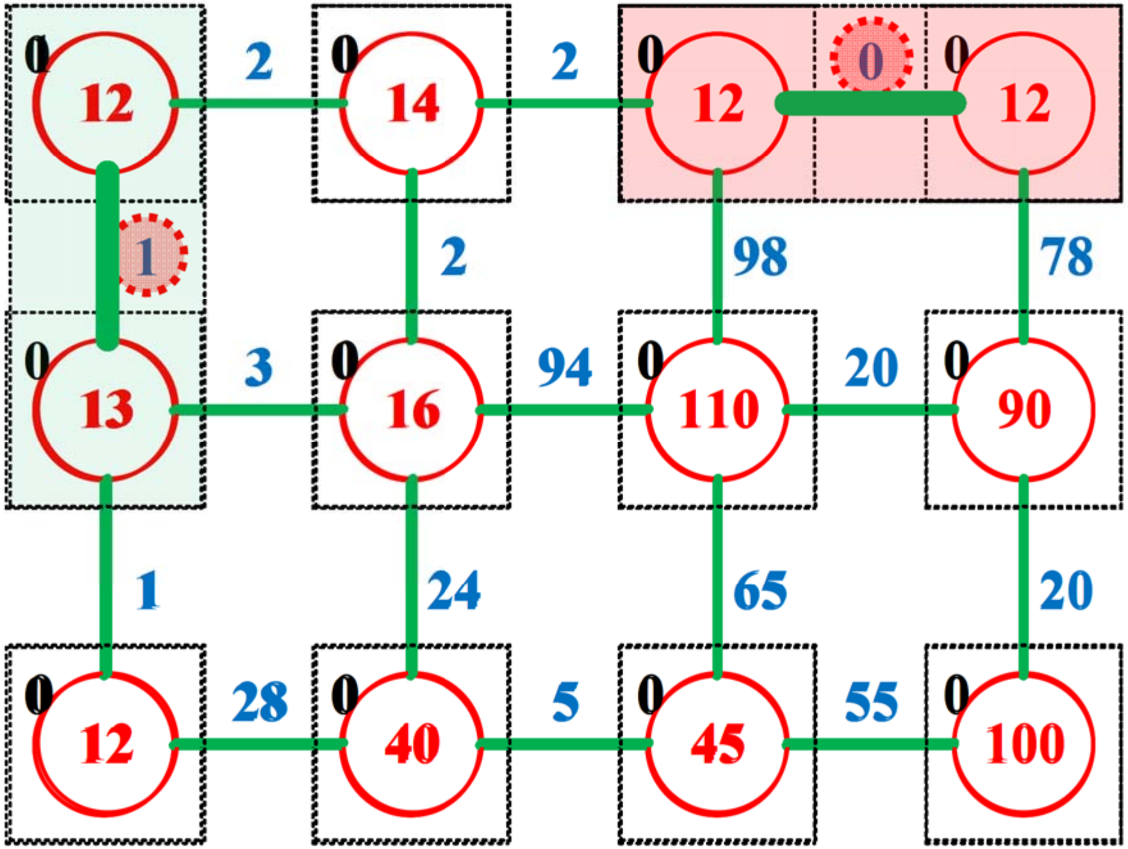
$$D(C_i, C_j) = false \text{ if } \begin{cases} Dif(C_i, C_j) \leq Int(C_i) + \frac{k}{\tau(C_i)} \\ Dif(C_i, C_j) \leq Int(C_j) + \frac{k}{\tau(C_j)} \end{cases}$$

$$D(C_i, C_j) = false \text{ if } \begin{cases} w(e_q) \leq Int(C_i) + \frac{k}{\tau(C_i)} \\ w(e_q) \leq Int(C_j) + \frac{k}{\tau(C_j)} \end{cases}$$

merge condition

Segmentation Algorithm

- Tiny Example ($k = 100$)



$$\begin{cases} w(e_q) \le Int(C_i) + \dfrac{k}{\tau(C_i)} \\ w(e_q) \le Int(C_j) + \dfrac{k}{\tau(C_j)} \end{cases}$$

$$0 \le 0 + \frac{100}{1} = 100$$

$$0 \le 0 + \frac{100}{1} = 100$$

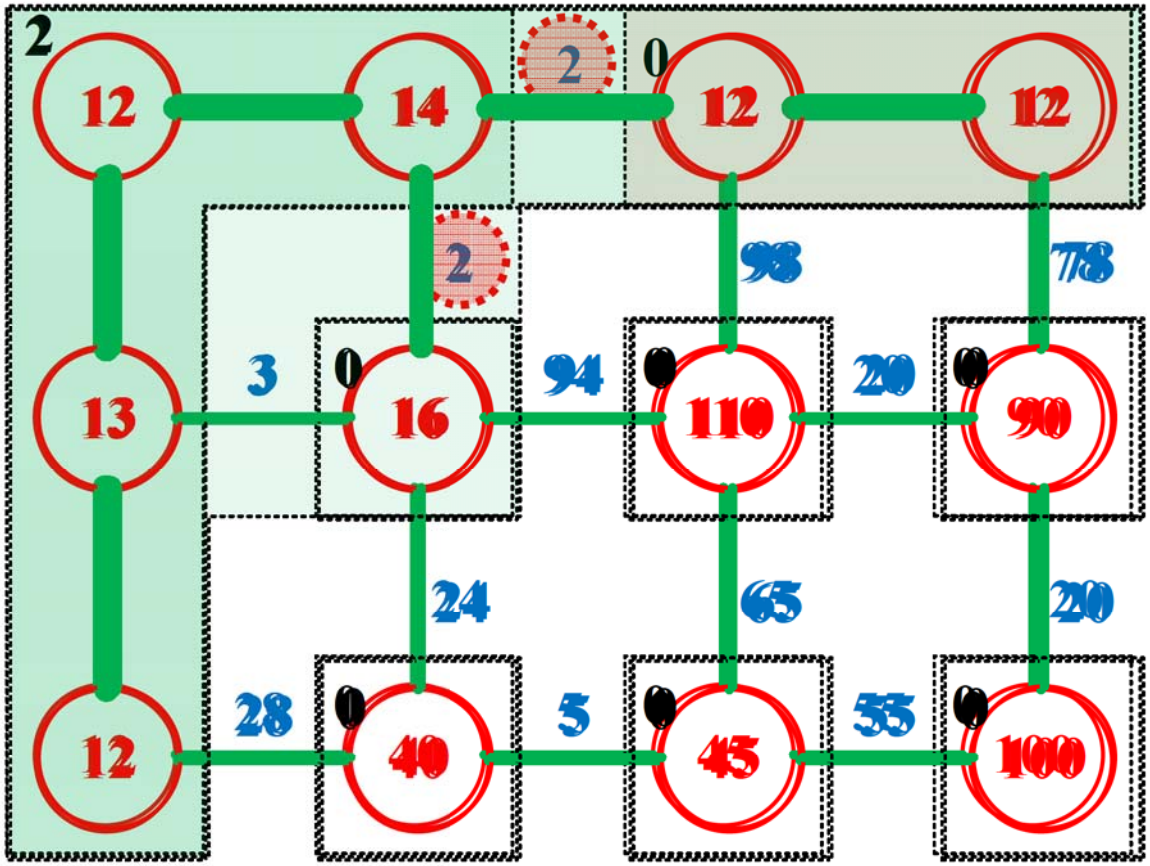
$$1 \le 0 + \frac{100}{1} = 100$$

$$1 \le 0 + \frac{100}{1} = 100$$

Segmentation Algorithm

- Tiny Example ($k = 100$)

$$\begin{cases} w(e_q) \le Int(C_i) + \dfrac{k}{\tau(C_i)} \\ w(e_q) \le Int(C_j) + \dfrac{k}{\tau(C_j)} \end{cases}$$



$$2 \le 2 + \frac{100}{4} = 27$$

$$2 \le 0 + \frac{100}{2} = 50$$

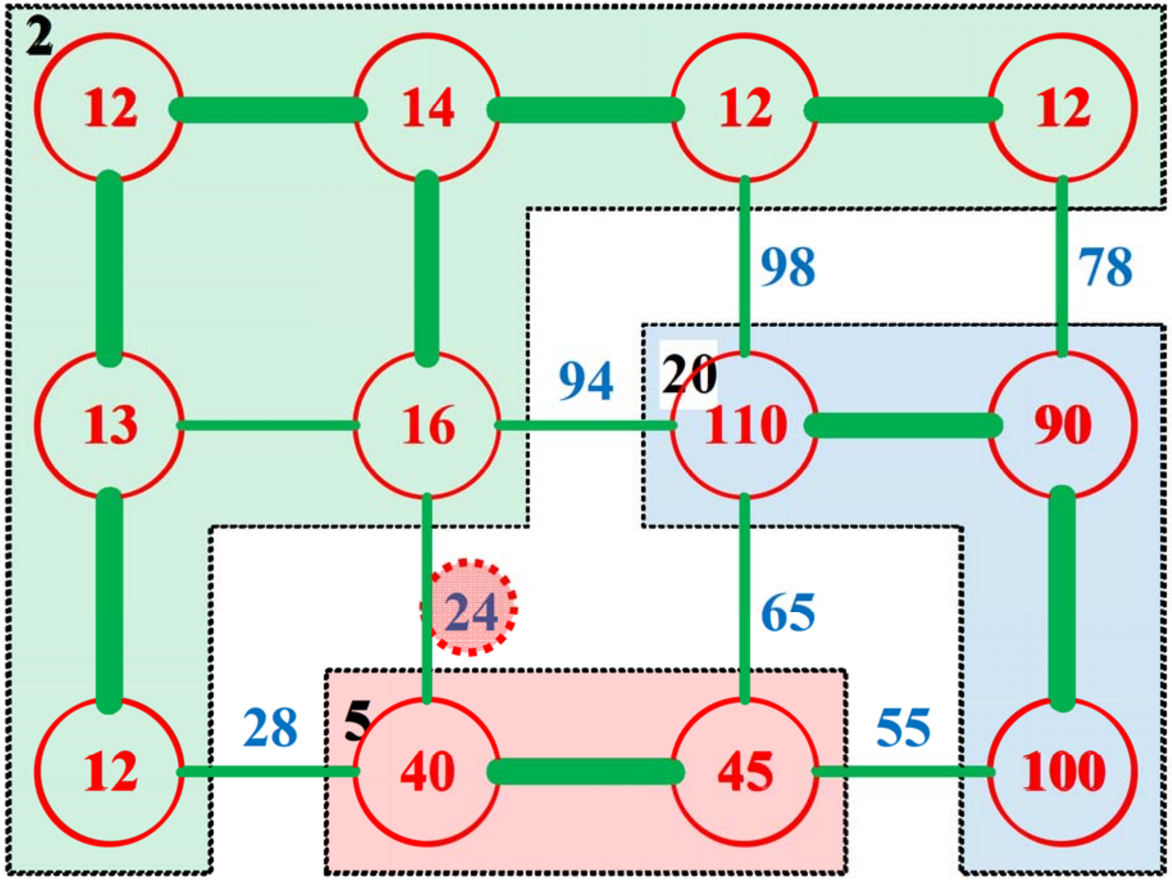
$$2 \le 2 + \frac{100}{6} = 18.67$$

$$2 \le 0 + \frac{100}{1} = 100$$

Segmentation Algorithm

- Tiny Example ($k = 100$)

$$\begin{cases} w(e_q) \le Int(C_i) + \dfrac{k}{\tau(C_i)} \\ w(e_q) \le Int(C_j) + \dfrac{k}{\tau(C_j)} \end{cases}$$



$$24 > 2 + \frac{100}{7} = 16.29$$

$$24 \le 5 + \frac{100}{2} = 55$$


thank you
© 1996, 2002 SANRIO CO., LTD.