Optimal Substructure Property

- Thus $B(T) = B(T') + (f(x) + f(y))$
- If T' represents a nonoptimal code, then there exists a tree T'' such that $B(T'') < B(T')$
- Since z is treated as a character in C' it appears as a leaf in T''.
- We can turn z in T'' as an internal node with x and y as its children then we get code for C with cost

  $$B(T'') + f(x) + f(y) \leq B(T)$$

  which contradicts optimality of T.
- Thus T' must be optimal.

©R. K. Ghosh

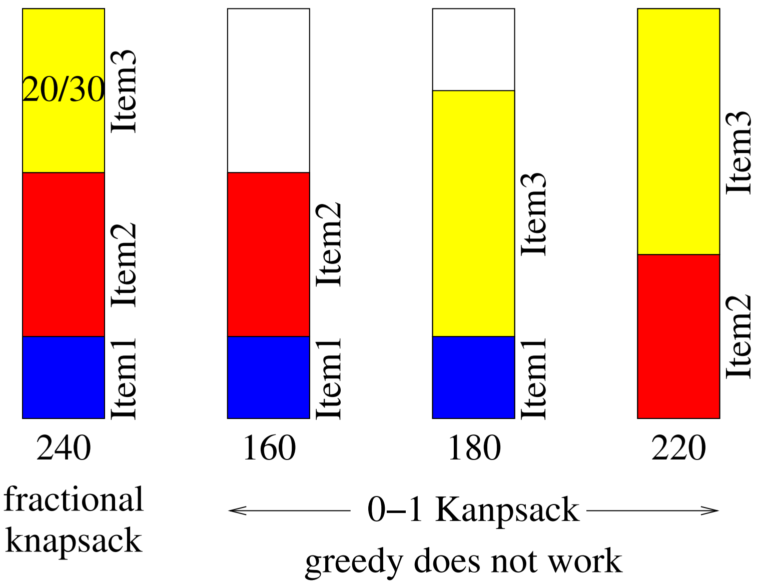
Knapsack Problem

- There are two variations of knapsack problem.
  - 0-1 knapsack problem.
  - Fractional knapsack problem.
- 0-1 knapsack problem cannot be solved by greedy algorithm.
  - The thief robbing a store can either take an item in full or must leave it entirely.
  - Therefore, it is possible that the thief may have to leave a set of other items whose combined value would be greater than what he could choose by the greedy approach.
- Fractional knapsack can be solved using greedy.
  - Thief can take item with the maximum per unit cost among the remaining as much as capacity of knapsack allows.

Knapsack Problem

- Let capacity knapsack be $W = 50kg$. Let there be 3 items.

Item	Weight	Value
1	10	60
2	20	100
3	30	120



Optimal Substructure Property

- Let W be the weight of most valueable load.
- Now suppose we remove a weight w for item i from it then
- $W - w$ must contain optimal selection for other $n - 1$ item, and the remaining $w_i - w$ weight of item i. Otherwise the initial value is not optimal.

©R. K. Ghosh

Greedy Choice Property

- Let v_h, w_h relates to item with highest value per unit weight.
- Let $L(i)$ represent weight of ith item in thief's loot L.
- As much as possible item with highest value per unit must be included.
- Total value of the loot

$$\sum_i L(i) \cdot \frac{v_i}{w_i}$$

  where v_i is the total value of item i with total weight w_i.
- If some of item h is left and $L(j) \neq 0$ for some $j \neq h$, then we can replace item j with h to get a higher value.
- Because $L(j) \cdot \frac{v_j}{w_j} \leq L(j) \cdot \frac{v_h}{w_h}$

Greedy Choice Property

- First case: if $W < w_h$
  - $L(h) = W$ and $L(j) = 0$ for all $j \neq h$ is the only solution.
- Case 2: some capacity be left after greedy choice. $L(i), L(j) > 0$. Assume i is the first choice and $j = h$ is a subsequent choice.
  - $V = L(i)\frac{v_i}{w_i} + L(h)\frac{v_h}{w_h}$
  - If we choose $L(h)$ of item h first and $L(i)$ of item i next, the value

$$V' = L(h)\frac{v_h}{w_h} + L(i)\frac{v_i}{w_i}$$

  - Thus $V' = V$.

Greedy Choice Property

- We have proved that as much as possible highest valued item is to be taken
- Also we proved that any sequence choices for optimal value solution can yield a corresponding sequence with greedy choice.

©R. K. Ghosh

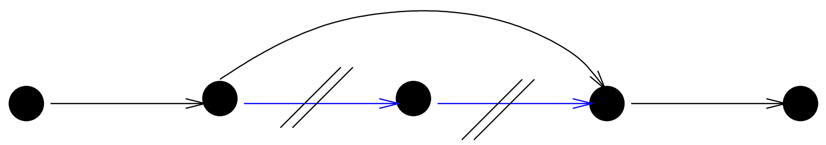
Shortest Path

- We already know Dijkstra's algorithm for shortest path.
- Lets us prove that it is greedy. We need proves for
  - Optimal substructure property.
  - Greedy choice property.

©R. K. Ghosh

Shortest Path

- Optimal substructure property is immediate from **cut-and-paste** argument.



- We can create shorter path by cutting sub-optimal subpath from a given path.
- So if a path is known to be optimal then it has optimal subtructure.

©R. K. Ghosh