

Design and Analysis of Algorithms

L17: Greedy Algorithms

Dr. Ram P Rustagi
Sem IV (2020-Even)
Dept of CSE, KSIT
rprustagi@ksit.edu.in

Resources

- Text book 2: Sec 4.1, 4.3, 4.4
- Text book 1: Sec 9.1-5.4 - Levitin
- RI: Introduction to Algorithms
 - Cormen et al.

Overview: Greedy Algorithms

- Basis of greedy algorithm:
 - Make the choice that seems best at the moment.
- Basics of Greedy Algorithms
 - A paradigm that build solutions using one piece at a time
 - Choose the next piece that is most obvious and provides immediate gain, i.e.
 - Maximizes benefit or minimizes cost
 - Expects that such local optimal solutions may lead to global optimal solution,

Greedy Algorithms

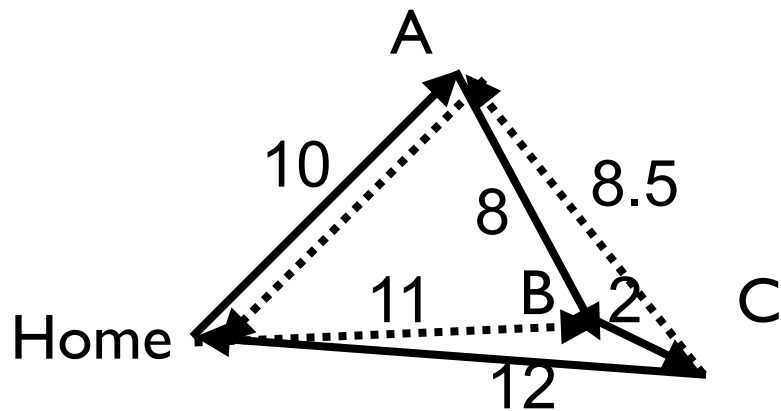
- How to decide which choice is optimal
 - Define an objective function and optimize the same with the choice to be made
 - Repeat the process at each step.
 - There is no going back to reverse the decision
- Advantages:
 - Easy to design a greedy algo (there can be multiple)
 - Complexity time analysis is comparatively easier
 - For divide-n-conquer it may not be easy
 - Depends on number of sub-problems and size
- Disadvantages:
 - How to ensure that chosen algorithm is correct

Coin Change Problem

- Issue minimum number of coins for a given value
- Amount to be dispensed: Rs 43
 - Consider coin denomination: Rs 1, 2, 5, 10, 20
 - 2 coins of Rs 20
 - 1 coin of Rs 2, and Rs 1
 - » Total coins: $2 + 1 + 1 = 4$
 - Consider coin denomination as: Rs 1, 2, 5, 10, 20, 25
 - 1 coin of Rs 25, Rs 10, Rs 5, Rs 2, Rs 1
 - » Total coins : 5
 - Optimal case (4 coins)
 - » 2 coins of Rs 20,
 - » 1 coin of Rs 2, and Rs 1
- Q: What is the Greedy approach, Objective Function?

Visting Friend to Invite

- Visiting friends to invite for a party:
 - May not be possible to visit all friends in a day
- Greedy approach?:
 - Start from visiting the nearest friend.
 - Criteria for nearest:
 - Objective function?



Greedy approach: HABCH
Distance = $10+8+2+12=32$

Non-Greedy approach: HBCAH
Distance = $11+2+8.5+10=31.5$

Greedy Algorithm

- Subset paradigm: Approach
- Consider the input in an order
 - Determined by some selection procedure
- If the selected input leads to feasible solution
 - That input is added to the solution
- Else
 - Do not consider that input
- Selection procedure is based on some optimization measure
 - The measure could be the objective function
 - There may several optimization measures possible

Greedy Algorithm

- Approach: subset paradigm

```
SolType Greedy(a[], n) {  
    //a[1:n] contains the n inputs  
    SolType solution = EMPTY // initialize  
    for i=1 to n do  
        Type x = Select(a)  
        if Feasible(solution, x)  
            solution = Union(solution, x)  
    // end for  
    return solution  
} // end algo
```


Subset Paradigm: Analysis

- Function `Select()` selects an input from `a []` and removes it.
- The selected input's value is assigned to `x`
- `Feasible` is a boolean valued function
 - Determines whether `x` can be included or not
- `Union()` combines `x` with the solution and updates the objective function
- Method `Greedy` describes how a typical greedy algorithm works

Coin Change Problem: Analysis

- Assumption: Enough number of coins for each denomination are available
- Initial solution: `empty` `Change`
- At each stage, 1 coin is selected and
 - added to `Change`
- Coin is selected using greedy criteria
 - It should increase total amount of `Change` as much as possible
- `Feasible` function:
 - `Change` `given` must equal the total amount
 - `Change` should not exceed the total amount

Algo: Coin Change

```
// D[1:n] # coin denominations in ascending order e.g.  
// D=[1, 2, 5, 10, 20, 50, 100]  
// res[1:n] # number of coins required, initially zero  
// Amt:# amount to be returned  
Algo CoinChange (amt)  
    index=n # start from highest coin  
    while (index>0)  
        while (amt >= D[index])  
            res[index] = res[index] + 1  
            amt = amt - D[index]  
            index=index-1  
//invocation of algo  
CoinChange (252)  
print (res) # print the count count
```

Max Water to Households

- A colony has number of houses and they get water from a water tanker of capacity T .
 - Each household H_i has a container having size C_i .
 - You are a politician and would like to oblige max number of households.
 - Each house gets a full container not partially filled.
 - At the same time would like to keep minimum water in tanker after filling the containers since that becomes wasted efforts. Can we solve this using greedy algorithm.
- Example:
 - $T=150L$, $C_1=50L$, $C_2=60L$, $C_3=115L$
 -

Machine Scheduling Problem

- Given
 - N tasks and infinite supply of machines on which these tasks can be run.
 - Each task has a start time s_i and finish time $f_i > s_i$. $[s_i, f_i]$ is called processing interval.
 - Two tasks overlap if their intervals overlap at a point other than start time and end time.
- Problem: Find optimal number of machines on which these tasks can be assigned.
- Feasible solution:
 - Assign one task to each machine. It is feasible but not optimal since this would need N machines.

Machine Scheduling Problem

- Example: 7 tasks with their start and end times.

Task	A	B	C	D	E	F	G
Start time	0	3	4	9	7	1	6
Finish	2	7	7	11	10	5	8

- Q: What should be the Greedy approach for machine assignment
 - Define new machine: on which task is run 1st time.
 - Define old machine: on which some task is already completed
- Approach: Use old machine for next task if available, else use new machine.

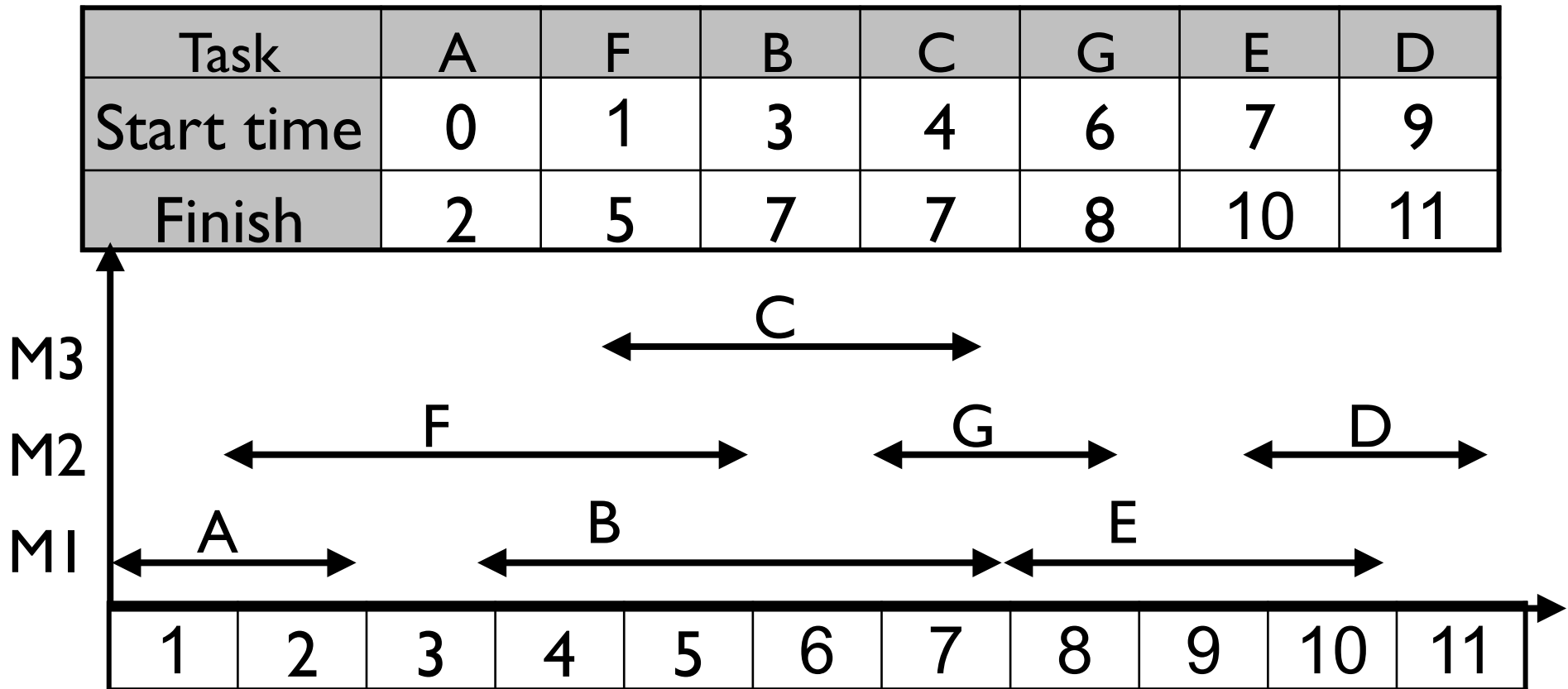
Machine Assignment

- Sort the tasks as per start time

Task	A	B	C	D	E	F	G
Start time	0	3	4	9	7	1	6
Finish	2	7	7	11	10	5	8

Task	A	F	B	C	G	E	D
Start time	0	1	3	4	6	7	9
Finish	2	5	7	7	8	10	11

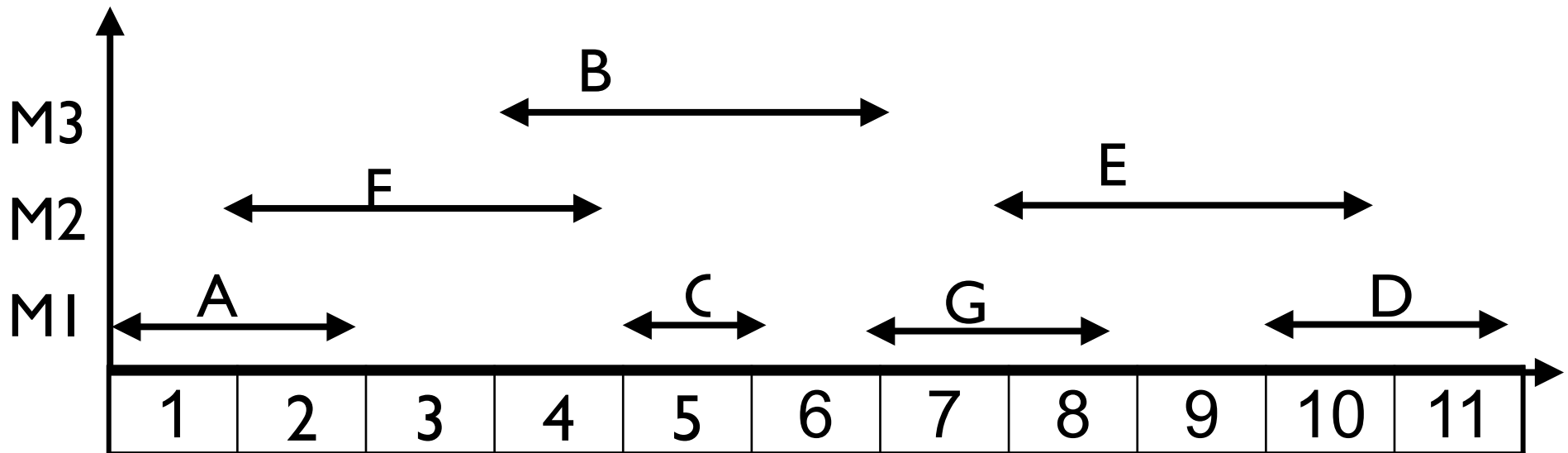
Machine Assignment: sort S.T.



Q: If we sort the task as per finish time, will it work?
Which one would you prefer sort by start time or finish time?

Machine Assignment: sort E.T.

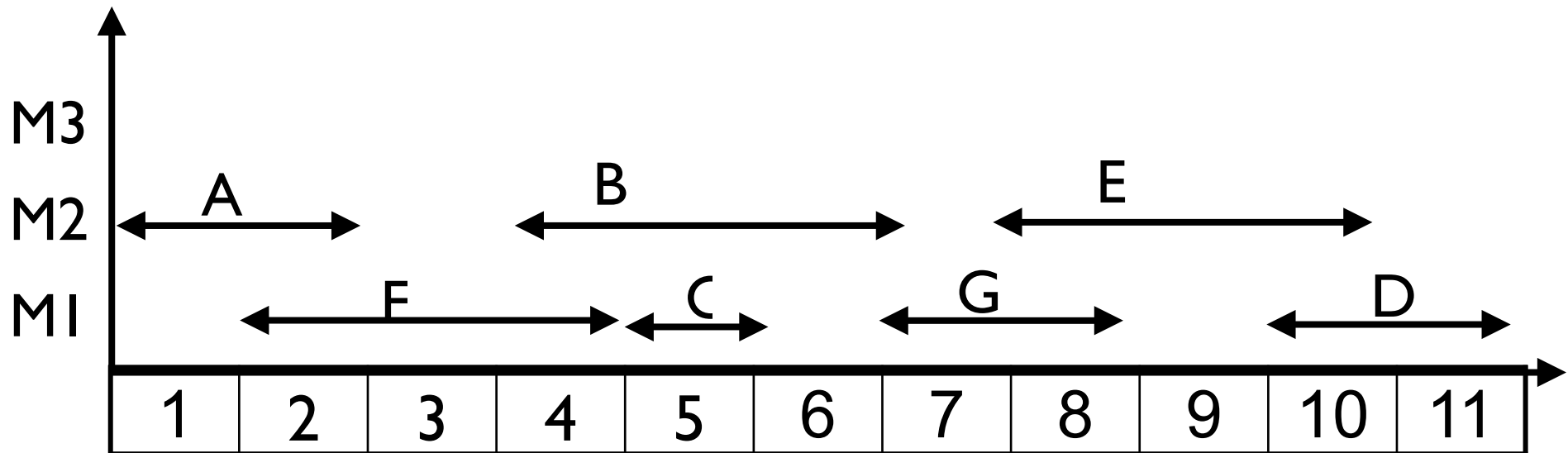
Task	A	F	C	B	G	E	D
Start time	0	1	4	3	6	7	9
Finish	2	4	5	6	8	10	11



Q: If we sort the task as per finish time, will it work?
Which one would you prefer schedule by start time or finish time?

Machine Assignment: sort E.T.

Task	A	F	C	B	G	E	D
Start time	0	1	4	3	6	7	9
Finish	2	4	5	6	8	10	11



Sort the task as per finish time, will it work?
Schedule by finish time decreasing order?

Algo: Machine Assignment

- Write algo for machine assignment

Exercise

- Consider the machine assignment problem but with only one machine.
- Problem: Find the largest number of tasks that can be assigned to this machine.
- What should be the greedy approach?

Summary

- Greedy algorithm
- Coin Change problem
- Machine Task problem
- Water container fulfillment problem