v0.1: Seth Gilbert
V1.4: Steven Halim

CS4234
Optimiz(s)ation Algorithms

L7 – (Weighted) **Max-Cardinality -(Bipartite)-Matching**

https://visualgo.net/en/matching

CS3233/CS4234
Dual Slides ☺

Dr. Steven Halim

Roadmap (for CS4234) – Week 07

Graph Matching

- Overview
- Greedy Bipartite Matching (Special Case)
- Unweighted MCBM: Max Flow (Dinic's) review, Augmenting Path, Hopcroft-Karp (nice *theoretical* result), Augmenting Path++
- Hall's Marriage Theorem
- A few other relevant applications will be discussed in T07 ☺
  - Remember that T06 is going to be just past paper discussions

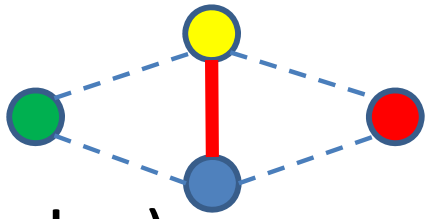
Roadmap (for CS4234) – Week 09

Graph Matching

- Weighted MCBM: (Min/Max) Cost (Max) Flow, Hungarian (Kuhn-Munkres algorithm)
- Unweighted MCM: Edmonds' Matching (**overview**)
- Weighted MCM: DP with Bitmask **(small graph only, review…)**
  - This DP with Bitmask solution will also solve other variants, but only if they are posed on small ($V \leq 20$) graphs…
  - Still unable to make it work for Christofides's 1.5-Approximation algorithm for M-R/NR-TSP as of year 2020

Graph Matching

A matching (marriage) in a graph **G = (V, E)** (real life) is a subset **M** of **E** edges in **G** (special relationships) such that no two of which meet at a common vertex (no affair)

Thus a.  and b. 

are matchings (red thick edge),

But trying to match both edges in c. 

is invalid since there is an overlapping vertex

Max-Cardinality-Matching (MCM)

Usually, the problem asked in graph matching is the size (cardinality) of a maximum (not maximal) matching

A maximum matching is a matching that contains the largest possible number of edges

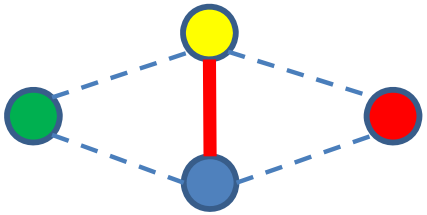
A possible variant: Perfect Matching

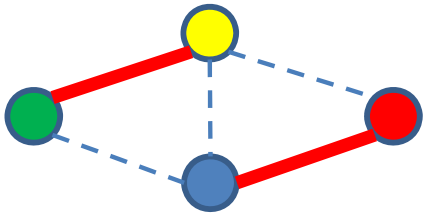
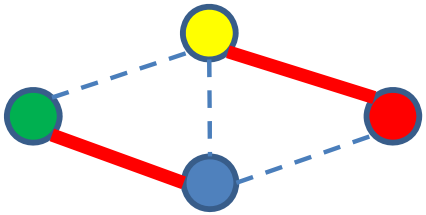
- MCM with no vertex left unmatched
  - PS: This is *impossible* on graph with odd number of vertices

Examples

 is a max(imum) matching (0 matching)
(no edge to be matched)

 is also a max(imum) matching (1 matching)
(no other edge to be matched)

But  is not a max(imum) matching
(it is a max(**imal**) matching btw)

as we can change it to  or 

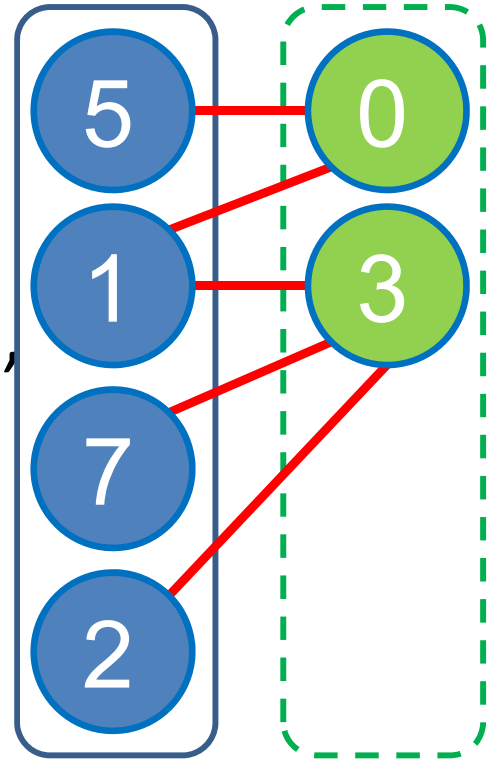
(2 matchings)

Max-Cardinality-Bipartite-Matching (MCBM)

A **Bipartite** Graph is a graph whose vertices can be divided into two disjoint sets **X** and **Y** such that every edge can only connect a vertex in **X** to one in **Y**

Matching in this kind of graph is **a lot easier** than matching in general graph

So if we are given a graph matching problem, **our first check** is to see whether the given input graph is bipartite…

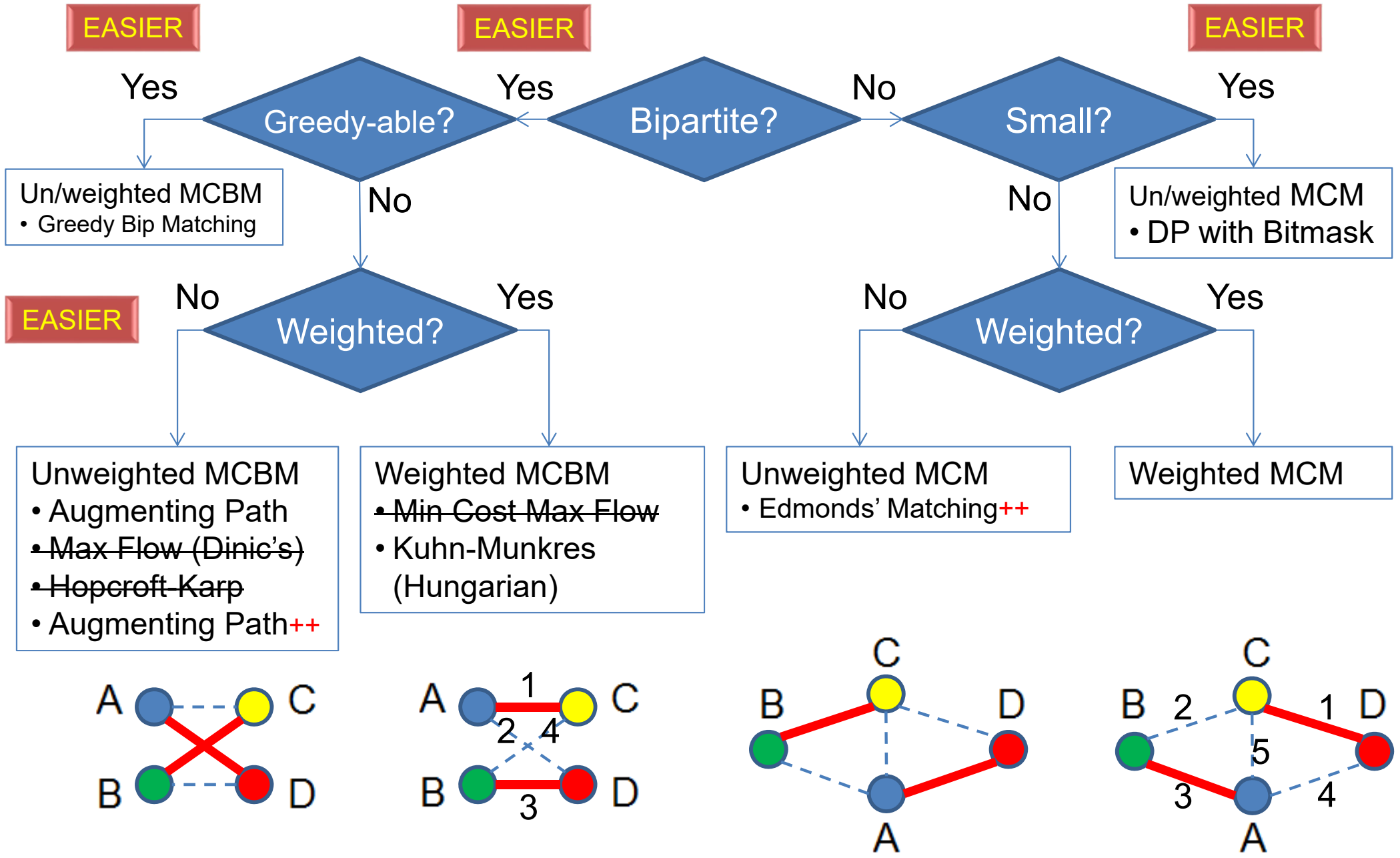


Weighted Matching

In some applications, the weights of edges are not uniform (1 unit) but varies

We may then want to take MCBM or MCM with minimum (or even maximum) total weight

Types of Graph Matching



Greedy Bipartite Matching

- Not all MCBM problems require special graph matching algorithms
- Let's see https://nus.kattis.com/problems/l
- Can be modeled as MCBM
- But the problem constraints allows the usage of a Greedy solution
- CP4 Book 1, Section 3.4.1 page 159

Hidden, review the recording?

e-Lecture: https://visualgo.net/en/matching?slide=4 to slide 4-8

Solutions:

Augmenting Path Algorithm (very simple to implement)

~~Max Flow (Dinic's, longer to code)~~

~~Hopcroft-Karp Algorithm (essentially Dinic's too)~~

Augmenting Path Algorithm++ (the ++ is important :O)

UNWEIGHTED MCBM

e-Lecture: https://visualgo.net/en/matching?slide=4-1 to slide 4-3

Finding MCBM via

AUGMENTING PATH ALGORITHM CP4 BOOK 1 SECTION 4.6.3

See https://visualgo.net/en/maxflow, select modeling, Bipartite Matching, all ones

Time Complexity of such modeling:
Depends on the chosen Max Flow algorithm, **but much faster** than general case as the capacities are **all ones (unit weights)** and the graph **is bipartite**, e.g., O(**m^2**) for basic FF (discussed in Tut04), or
O(**sqrt(n)*m**) (other analysis: O(**min($n^{2/3}$, $m^{1/2}$)*m**)) for Dinic's

e-Lecture: https://visualgo.net/en/matching?slide=4-4

Finding MCBM by reducing this problem into

MAX FLOW
CP4 BOOK 2 SECTION 8.4 & 8.5

CP4 Book 2 Ex 8.5.3.1*: Why the graph has to be directed?

Answer hidden, review the recording?

When To Use Max Flow Solution?

Only if we are solving Bipartite Matching
with Capacity (the "Assignment Problem")

- e.g., UVa 00259, CP4 Book 2 Section 8.4.5

See https://visualgo.net/en/maxflow, select modeling, Bipartite Matching, random

- This variant will be **much slower** to solve if reduced to MCBM, imagine if the typical capacities of the edges are big, reducing the problem to MCBM will lead to a gigantic bipartite graph…

But if we are solving pure MCBM (capacities are all 1),
just use the Augmenting Path algorithm
(easier to code and faster)

e-Lecture: https://visualgo.net/en/matching?slide=4-5 to slide 4-6

Finding MCBM via

HOPCROFT-KARP ALGORITHM (CP4 BOOK 2 SECTION 9.26)

Hopcroft-Karp Algorithm (1973)

That is, up to √V iterations only

Hopcroft-Karp runs in O(√**V** * **E**), proof omitted

- For the extreme test case in previous slide, this is O(sqrt(**N**+**M**)*__N__*__M__)
- With **M** = **N**, this is about O(**N**$^{5/2}$), OK for **N** ≤ [500..600]
  - PS: Dinic's algorithm on Bipartite Matching graph **also runs** in O(N$^{5/2}$)

Is this algorithm **must be learned** in order to have a good MCBM algorithm that does well in practice$_{\text{/programming contest}}$?

https://github.com/stevenhalim/cpbook-code/blob/master/ch4/mcbm.cpp

e-Lecture: https://visualgo.net/en/matching?slide=4-7 to slide 4-8

Finding MCBM via

AUGMENTING PATH ALGORITHM++

CP4 BOOK 2 SECTION 8.5.3

Live Solve

https://nus.kattis.com/problems/b

Hidden, review the recording?

From my Augmenting Path Algorithm++ baseline code (C++)

Time permitting

THE SINGLE MOST IMPORTANT PART OF THIS LECTURE

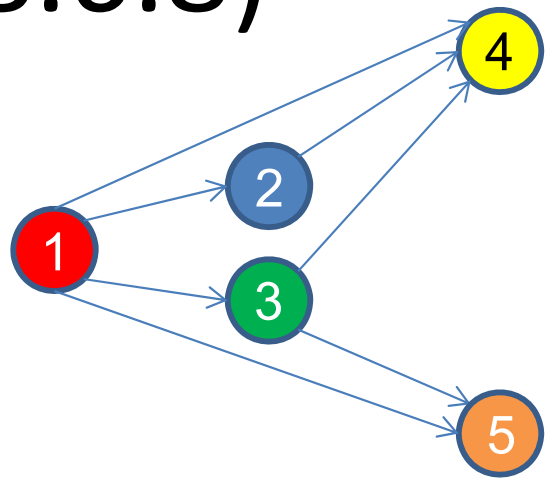
SO KEEP AN EYE ON LEFT/RIGHT, ROW/COL, alternate ROW/COL, PRIME/COPRIME, ODD/EVEN, MALE/FEMALE, JOB/EMPLOYEE, bi-coloring, out-degree only/in-degree only, no-odd-length-cycle, Tree*, etc (and lots of other common stuffs)
(see CP4 Exercise 4.6.3.2*)

MORE EXAMPLES OF MCBM PROBLEMS

Min Path Cover in DAG (CP4 Book 2 Section 8.6.8)

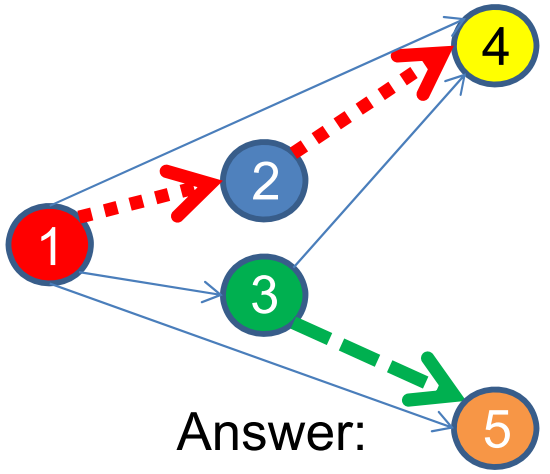
Illustration:

- Let **V** = passengers, we draw edge between two vertices if a single taxi can satisfy the demand of both passengers on time…
- What is the minimum number of taxis that must be deployed to serve all passengers?



This problem is called: Min Path Cover

- Set of directed paths s.t. every vertex in the graph belong to at least one path (including path of length 0, i.e., a single vertex)

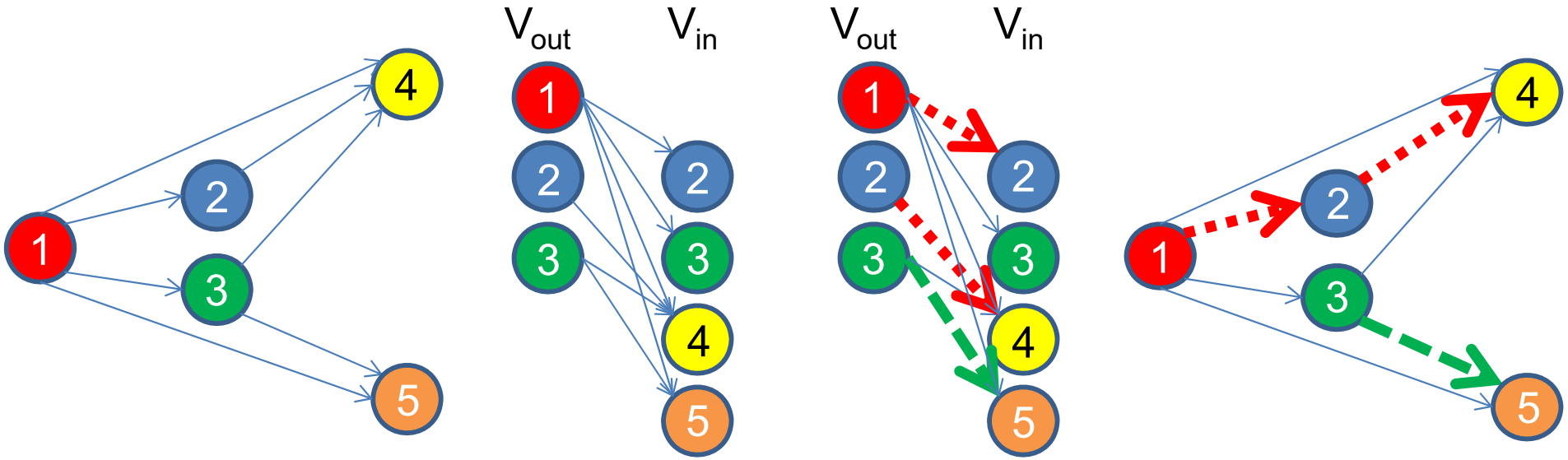


Answer:
2 Taxis!

Min Path Cover in DAG (CP4 Book 2 Section 8.6.8)

Also an NP-Hard optimization problem on general graph

- But on DAG, we can reduce this to MCBM!
  - Notice vertices with out-degrees only and in-degrees only

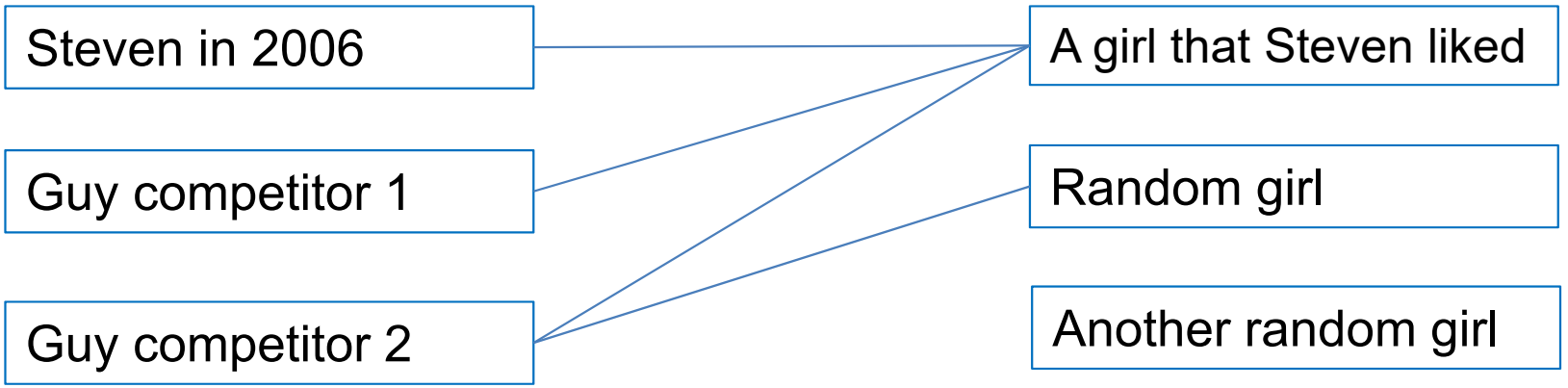


Note to self: Add this to "Modeling" section of VisuAlgo later and use VA screenshots instead

V = 5, initially we need 5 paths
M = MCBM = 3
each matching reduce the need of one path
V–**M** = 5–3 = 2 paths can cover all vertices

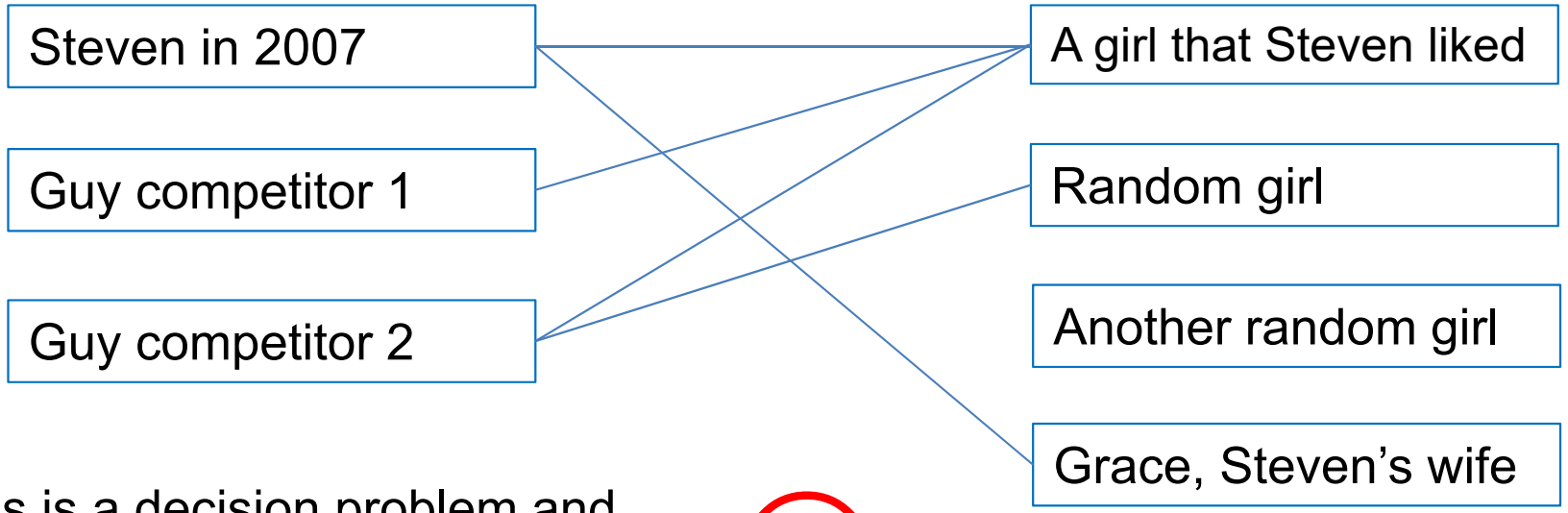
Hall's Marriage Theorem (1)

- Let **G** be a bipartite graph with bipartite sets **X** and **Y**
- Then there exists a matching that *covers **X***
  if and only if *for each subset **W*** of **X**, **|W| ≤ |N(W)|**



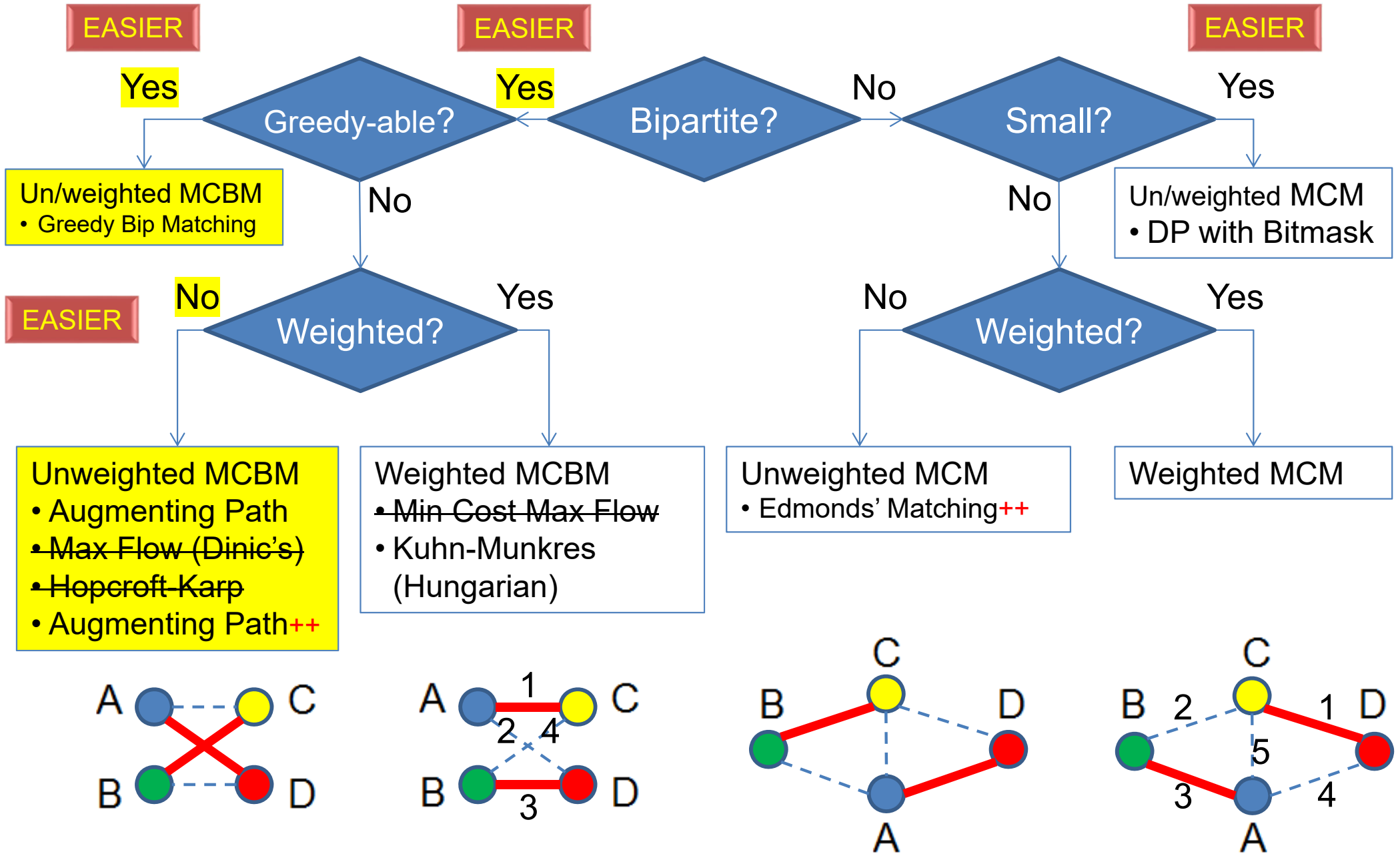
Hall's Marriage Theorem (2)

- Let **G** be a bipartite graph with bipartite sets **X** and **Y**
- Then there exists a matching that *covers **X***
  if and only if *for each subset **W*** of **X**, **|W| ≤ |N(W)|**



PS: This is a decision problem and
"each subset is pseudo-polynomial"

Summary (so far…)



References

- Mostly CP4, Book 1 Section 4.6.3 +
  Book 2 Section ~a bit of 8.4, 8.5, ~a bit 8.6
- TopCoder PrimePairs, RookAttack solution
- More matching exercises/applications during next lecture (Week 09) and T07 (Week 09)