# CS 6901 (Applied Algorithms) – Lecture 9

Antonina Kolokolova

October 18, 2016

## 1 Solving systems of difference constraints as detecting negative cycles

We talked a lot about graph problems as the main modeling tool. Another very common and general way to model a problem is as a system of linear equations.

For example, suppose that you have 4 jobs (or exams, etc) that need to be scheduled, possibly in parallel. Suppose that the length of the first job is 3, and of the second job is 5. Suppose that there are additional constraints like this: the second job has to start after the first finishes, the third has to start at least five minutes after the second ends, job number 4 has to start at least 4 minutes before job number 3, and job number 3 has to start no later than 7 minutes after the first one starts. This gives us the following system (here, $s_1, s_2, s_3, s_4$ are variables denoting starting times of jobs that we are trying to determine):

$$\begin{cases} s_1 + 3 \leq s_2 \\ s_2 + 10 \leq s_3 \\ s_4 \leq s_3 - 4 \\ s_3 \leq s_1 + 7 \end{cases} \tag{1}$$

Is it possible to schedule these jobs satisfying the constraints? In this case, no: to finish job 1, then do job 2, then wait 5 minutes would take time $3 + 5 + 5$, making it impossible to schedule job 3 at most 7 minutes after job 1 starts. So given a system of inequalities like this, how do we find out if a solution exists?

In general, if the solution is not required to be integers, it is possible to find it in polynomial time using linear programming techniques. However, the running time is not that great (e.g, $O(n^{3.5} * L)$, where $n$ is the number of variables and $L$ is the length of the encoding of the system). But in a special case, as we have above, where every inequality only contains two variables and a constant, it is possible to check if there is a solution in time $O(mn)$, where $m$ is the number of inequalities and $n$ is the number of variables. This class of inequalities is called difference constraints.

In order to solve this system, we represent it as a weighted directed graph (its "constraint graph"), with a property that there is a negative cycle in this graph if and only if the original system had no solutions. This graph is defined as follows. For every variable $x_i$ in the system, we introduce a vertex $v_i$, and for every equation, an edge. If an equation is $x_j \leq x_i + b_k$, then the corresponding edge will be $(v_i, v_j)$, with weight $b_k$. Note that the edges are directed, with the direction going from the variable on the "larger than" side to the variable on the "smaller than" side. Now, there is a negative cycle in this graph if and only if the system of equations was unsatisfiable: think about it as "$x_i$ is smaller than $x_{j_1}$ (plus something) which is smaller than $x_{j_2}$,... which is smaller than $x_i$ again". That would be a contradiction.

To simplify solving the system, we will add one more vertex $s$ to the constraint graph, with an edge of weight 0 from $s$ to each vertex in the graph. Thus, for the example above we get the following graph with the negative cycle $v_1, v_2, v_3$: