Algorithms – CS-37000

The car race problem

Instructor: László Babai   Ry-164
e-mail: laci@cs.uchicago.edu

The solution should be short, **elegant**, *and convincing.*

Let R be a subset of the $(n+1)^2$ points in the plane with integer coordinates between 0 and n. We call R the "race track." One of the points of R is designated as the start (S), another as the goal (G).

The points are represented as vectors (i, j). Cars are particles sitting on a point at any time. In one unit of time, a car can move from a point of R to another point of R, say from (i_1, j_1) to (i_2, j_2). The *speed vector* of the car during this time unit is defined as the vector $(i_2 - i_1, j_2 - j_1)$.

The *acceleration/deceleration* of the car is limited by the following constraint: from any one time unit to the next one, each coordinate of the speed vector can change by at most one.

For instance, if during time unit 6 the car was moving from point $(10, 13)$ to point $(16, 12)$ then its speed vector was $(6, -1)$ during this move; during the next time unit, the following are its possible speed vectors and corresponding destinations:

speed during time unit 7	location at the end of time unit 7
$(7, 0)$	$(23, 12)$
$(7, -1)$	$(23, 11)$
$(7, -2)$	$(23, 10)$
$(6, 0)$	$(22, 12)$
$(6, -1)$	$(22, 11)$
$(6, -2)$	$(22, 10)$
$(5, 0)$	$(21, 12)$
$(5, -1)$	$(21, 11)$
$(5, -2)$	$(21, 10)$

Of course only those locations are legal which belong to R (the car cannot leave the race track).

During time unit 0, the car rests at Start with speed $(0, 0)$. The objective is to decide whether or not the Goal is reachable at all and if so, to reach it using the minimum number of time units.

(a) Construct an example where the optimal route visits the same point 100 times (at different speeds).

(b) Find an optimal route in $O(|R| \cdot n^2)$ time. Describe your solution in clear English statements. Pseudocode not required. Algorithms discussed and analysed in class can be used as subroutines. Prove that your algorithm runs within the time claimed. *Hint.* Use BFS. The difficulty is in constructing the right graph to which to apply BFS. Do not overlook the possibility stated in (a).

(c) Solve the problem in $O(|R| \cdot n)$ time and space. (Note that you are not permitted to use an array with more than $O(|R| \cdot n)$ cells because of the space constraint.) (Hint: it is likely that you need only a minor modification to the algorithm you gave for (b) together with a more clever analysis.)