Walking Bus Challenge

F. Malucelli[1] E. Tresoldi[1]

[1]Dipartimento di Elettronica, Informazione e Bioingegneria - Politecnico di Milano
emanuele.tresoldi@polimi.it

January, 2017

Walking Bus: The Setting

A walking bus (*pedibus* in Italian) is a form of student transport for schoolchildren who, chaperoned by adults (usually a *driver* leads and a *conductor* follows), walk to school, in much the same way a school bus would drive them to school.

Like a traditional bus, walking buses have a fixed route with designated *bus stops* in which they pick up children.

Walking Bus: The Problem

An elementary school want to setup a walking bus system for its students.

Walking Bus: The Requirements

- The routes of the walking bus system must serve all students.
- A route starts form a bus stop, can go through other bus stops and ends at the school.
- All students on the route must be picked-up.
- Routes can merge but cannot split after merging.
- No student must travel more than *alpha* times the shortest path from his bus stop to the school.
- The design of the routes should ensure students' safety.

GOAL: minimize the number of chaperons involved.

Walking Bus: Mathematical Description

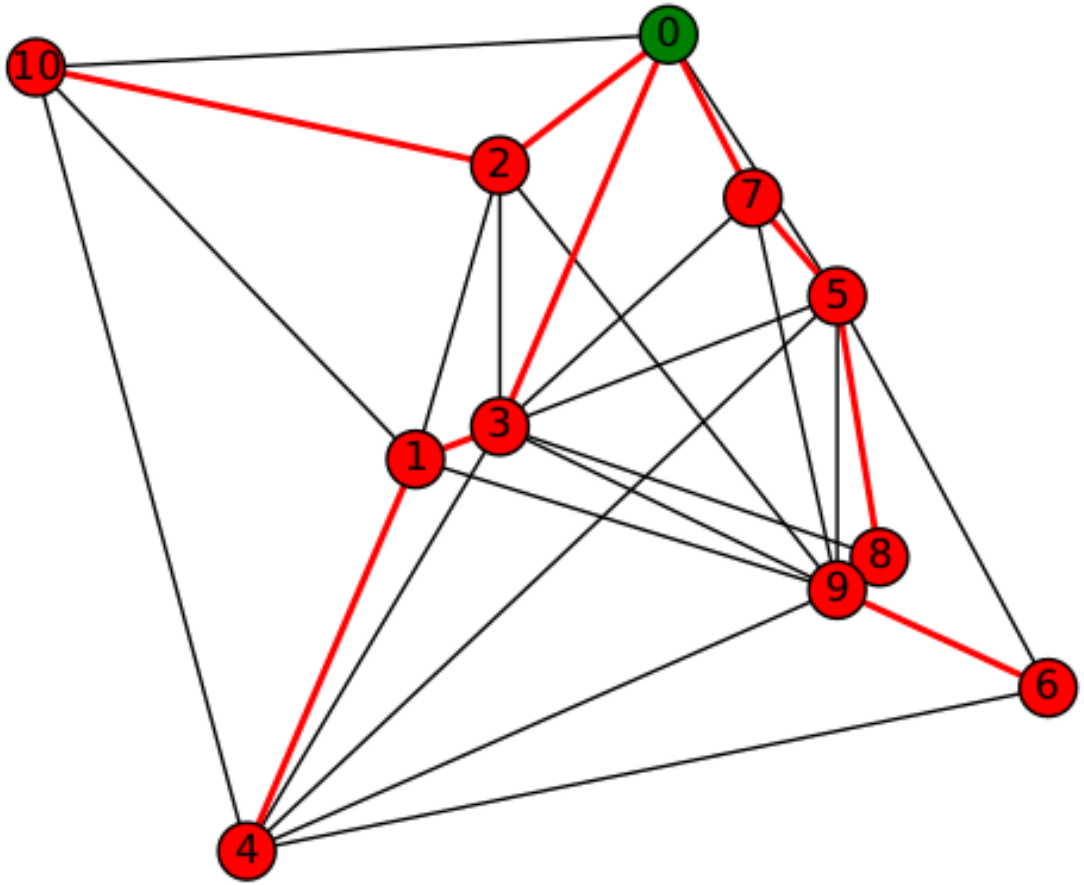
This problem can be seen as a special case of spanning tree problem:

- A complete graph $G = (N, A)$.
- $N = \{0..n\}$ is the set of nodes, node 0 is the school and all other nodes represent bus stops.
- Set $A = (i, j) : i \in N, j \in N$ is the set of arcs representing connections among nodes, a connection is represented by the shortest path from i to j.
- Parameter $c_{i,j}$: the length of the path from i to j.
- Parameter $d_{i,j}$: the risk (dangerousness) of the path from i to j.
- Parameter $\alpha > 1$: max percentage of the shortest path from bus stops to school students are willing to walk.

Primary GOAL find the *feasible* spanning tree, rooted in 0, for graph G with the minimum number of leaves, i.e. number of adults involved.

Secondary GOAL minimize the total risk.

Walking Bus: Solution Example



Walking bus solution example. Red nodes are the students, green node is the school. Red edges are the edges selected in the solution
(the graph is not complete).

Challenge Rules

- Build a **solver** for the walking bus problem.
- All solving approaches are accepted (mathematical models, exact algorithms and heuristic algorithms of any sort).
- Students can take part in the challenge in team, **max 3 students per team**.
- All students delivering a solver that is able to provide **non trivial solution** for the walking bus problem will receive **from 10 to 14 points** depending on the solver performance and are **exempt** from the lab exam and the second part of the written exam.
- **Deadline** for the submission of the solver: 24th of February 2017.
- In order to participate please **send an email** to emanuele.tresoldi@polimi.it.

What to Deliver

Participants must submit:

- *Complete source code* of the solver, the solver can be written in any of the following languages: AMPL, C/C++/C#, Java, Python. Any freely available library can be used. The solver must be compilable and runnable on either Windows 10 64 Bit or Linux 64 Bit. Execution time-limit: 1 hour (for each instance).
- *Instructions* on how to compile and run the solver.
- *Small report* describing the solution approach and results; a couple of pages should be enough.

All submissions must be sent to emanuele.tresoldi@polimi.it.

Evaluation

Solvers will be evaluated and ranked using a set of 10 instances.
Three different criteria will be considered for the evaluation.
From the most to the less important:

1. Number of leaves.
2. Risk.
3. Computational time (timelimit 1h per instance).

Beep

In the *Challenge* folder on *Beep* you can find:

- A set of 10 test instances to be used in order to implement end test the solver.
- A checker (a piece of software written in python 2.7) that can be used to check the feasibility of the solutions found.
- A document describing the problem and the rules of the challenge.
- A document describing all technical requirements for the solver and how to run the checker.