Support to ALGAV Practical assignment Sprint B

Deliveries Planning using Electrical Trucks

Curricular Unit: Algoritmia Avançada

Programme: Computer Engineering (ISEP)

If a problem is complicated, why not divide it in steps that make the solution simpler?

Once solved the step 1 problem, let us start with step 2 of this sprint, where we will look into the **constraints real batteries** will pose and that have an impact in the **total time** of the solution.

To help with this process we will use as a base a problem from a previous exam, dealing with an electric car tasked with small deliveries to the several schools belonging to the Polytechnical Institute.

Although bearing clear differences with the current work, this example may give you some ideas for the development of step 2, specially in what concerns the reasoning to use in order **to decide whether the truck batteries should be charged prior to leaving** for the next destination.

Let us reuse problems from last year's exam as "approximations" to the current one

Consider that the Polytechnical Institute wants to **optimize the time** consumed in **delivering paper documents** to a set of **several schools** (located in different places) using an **electric vehicle**.

That vehicle leaves the starting point (the main Institute building), passes by all the schools where documents must be delivered, and returns to the starting point.

The schools where deliveries must be made are represented by a Prolog fact ***deliveries_schools/1*** and the starting point is defined in the fact ***starting_point/1***, as per the following examples:

```
deliveries_schools([esmae,esmad-esht,ess,estg,iscap,isep]).

starting_point(ipp).
```

To support the use of electric cars, battery chargers were installed in every school as well as in the start/arrival point.

Suppose that the full battery charge takes 30 minutes and that the vehicle has an autonomy of 120 km. This knowledge is represented using the facts *chargeTime/1* and *maxAutonomy/1* as follows:

```
chargeTime(30).

maxAutonomy(120).
```

Let us reuse problems from last year's exam as "approximations" to the current one

The distances and travel times between each pair of schools (including the start/arrival point) are represented by the facts ***distTime(S1,S2,Dist,Time)*** where *S1* and *S2* are schools or the start/arrival point, *Dist* represents the distance between S1 and S2 in Km and *Time* the travel time in minutes.

Suppose that the times and distances are the same in both directions and are codified as follows:

```
distTime(ipp,esmae,4,20).
distTime(ipp,isep,2,5).
distTime(ipp,iscap,3,10).
distTime(ipp,ess,2,5).
distTime(ipp,estg,58,45).
distTime(ipp,esmad-esht,31,38).
distTime(esmae,isep,5,20).
distTime(esmae,iscap,6,25).
distTime(esmae,ess,4,20).
distTime(esmae,estg,60,55).
```

```
distTime(esmae,esmad-esht,34,48).
distTime(isep,iscap,5,10).
distTime(isep,ess,1,2).
distTime(isep,estg,59,45).
distTime(isep,esmad-esht,30,36).
distTime(iscap,ess,5,10).
distTime(iscap,estg,60,47).
distTime(iscap,esmad-esht,32,38).
distTime(ess,estg,59,45).
distTime(ess,esmad-esht,30,36).
distTime(estg,esmad-esht,70,55).
```

Let us reuse problems from last year's exam as "approximations" to the current one

a) [50%] Write the predicate ***calculate_cost(LS, Time, LCharging)*** that, for a given list of schools where documents must be delivered, returns the travel Time and a list ***LCharging*** of all the schools where the car battery had to be charged. Suppose that the car leaves the starting point with the battery fully charged (maximum autonomy). The predicate could be called as follows:

?- calculate_cost([esmae,esmad-esht,ess,estg,iscap,isep],Time, LCharging).

Time = 271,

LCharging = [ess, iscap]

Suggestions:

- remember that you have to include the start point at the beginning and end of the list of schools where deliveries have to be made.

- you will have to recharge the battery anytime the next step of the route has a distance greater than the car's autonomy in that moment. Therefore, you will be forced to increment the total trip time with the number of minutes specified in ***chargeTime***/1. The car's autonomy will be increased to the maximum value as per ***maxAutonomy***/1.

Let us reuse problems from last year’s exam as “approximations” to the current one

```
%a calculate_cost
calculate_cost(LS,Time,LCharging):-
    starting_point(SP),
    append([SP|LS],[SP],LScomplete),
    maxAutonomy(Autonomy),
    cost(LScomplete, Autonomy, Time, LCharging).

cost([_],_,0,[]).
cost([S1,S2|LS], Autonomy, Time, LCharging):-
   (distTime(S1,S2,Dist,T);distTime(S2,S1,Dist,T)),
   ((Autonomy<Dist,!, chargeTime(TCharge), maxAutonomy(A), A1 is A-Dist, LCharging = [S1|LPreviousCharges]);
    (A1 is Autonomy-Dist, TCharge is 0, LCharging = LPreviousCharges)),
   cost([S2|LS],A1,Time1,LPreviousCharges),
   Time is TCharge+T+Time1.
```

Let us reuse problems from last year's exam as "approximations" to the current one

b)[50%] Write a predicate ***min_time_seq(LS,LCharging,Time)*** that finds the sequence of schools visited for document deliveries that is made in the **lesser time**, using the calculation of the previous problem, and returns in ***LS*** the list with that best sequence, in ***LCharging*** the list with the schools where the recharge operations had to be performed, and in ***Time*** the total time of that best route solution (including charging times). The schools where deliveries must be made are specified in the list contained in the fact **deliveries_schools**/1.

For example, the call of the predicate could be the following:

?- min_time_seq(LS,LCharging,Time).

LS = [esmae, estg, esmad-esht, iscap, ess, isep],

LCharging = [estg],

Time = 215

Suggestions:

- you can generate all different schools sequences by performing the permutation of the list elements using the standard SWI Prolog predicate *permutation(ListOfSchools,ListOfSchoolsPermutated)*.

- store in a dynamic fact ***minTime*(*ListOfSchoolsPermutated, ListChargingSchools, Time)*** the *ListOfSchoolsPermutated* that has the smaller *Time* until the moment, while the times of the several available permutations are being calculated. The associated *ListChargingSchools* should also be included.

Let us reuse problems from last year's exam as "approximations" to the current one

```
%b min_time_seq
min_time_seq(LS,LCharging,Time):-(run;true),minTime(LS,LCharging,Time).
run:-
    retractall(minTime(_,_,_)),
    assertz(minTime(_,_,100000)),
    deliveries_schools(LSchools),
    permutation(LSchools,LSchoolsPerm),
    calculate_cost(LSchoolsPerm,Time,LCharging),
    update(LSchoolsPerm,Time,LCharging),
    fail.

update(LSPerm,Time,LCharging):-
  minTime(_,_,MinTime),
  ((Time<MinTime,!, retract(minTime(_,_,_)), assertz(minTime(LEPerm,LCharging,Time)),
    write('Time='),write(Time),  write(' '),write(LSPerm), write(' with recharges at '),write(LCharging),nl)
    ;true).
```

% writes just to check the lesser time update

What are the differences of our problem?

The exam problem doesn’t consider the **energy of the batteries** (in kWh). In the ALGAV practical assignment we will have to take into account the concept of **vehicle battery autonomy** (specified in Km).

We will assume that the **full charging time** is fixed and that the **distance** and **times** between two cities will be the same **regardless of the direction**.

Our concern will be to **check** at every stop whether the **current vehicle autonomy is enough** for it to arrive at the next stop. In our case it will have the added complexity of the **energy** (kWh) used depending on the **load** transported by the truck. One or more intermediate recharges of the batteries may be needed in order to complete the route, even if they are charged to 80% prior to leaving.

Work to be performed during the 2nd week of Sprint B:

- Adapt the solution found in the step 1 and add the treatment of the constraints posed by the battery recharging that will impact the total times of the different routes.

Calculation of the solution time

To understand what happens, we will make the manual calculations of a specific solution, i.e, a sequence of deliveries.

For the first example of a set of deliveries (5 deliveries) we will have the following facts:

```
%entrega(<idEntrega>,<data>,<massaEntrega>,<armazemEntrega>,<tempoColoc>,<tempoRet>)
%delivery(<deliveryId>,<date>,<deliveryLoad>,<deliveryWarehouse>,<loadTime>,<loadPickupTime>)
entrega(4439, 20221205, 200, 1, 8, 10).
entrega(4438, 20221205, 150, 9, 7, 9).
entrega(4445, 20221205, 100, 3, 5, 7).
entrega(4443, 20221205, 120, 8, 6, 8).
entrega(4449, 20221205, 300, 11, 15, 20).
```

The first permutation generated will be [1,9,3,8,11] (the second [1,9,3,11,8] and the last, the 32nd, would be [11,8,3,9,1]).

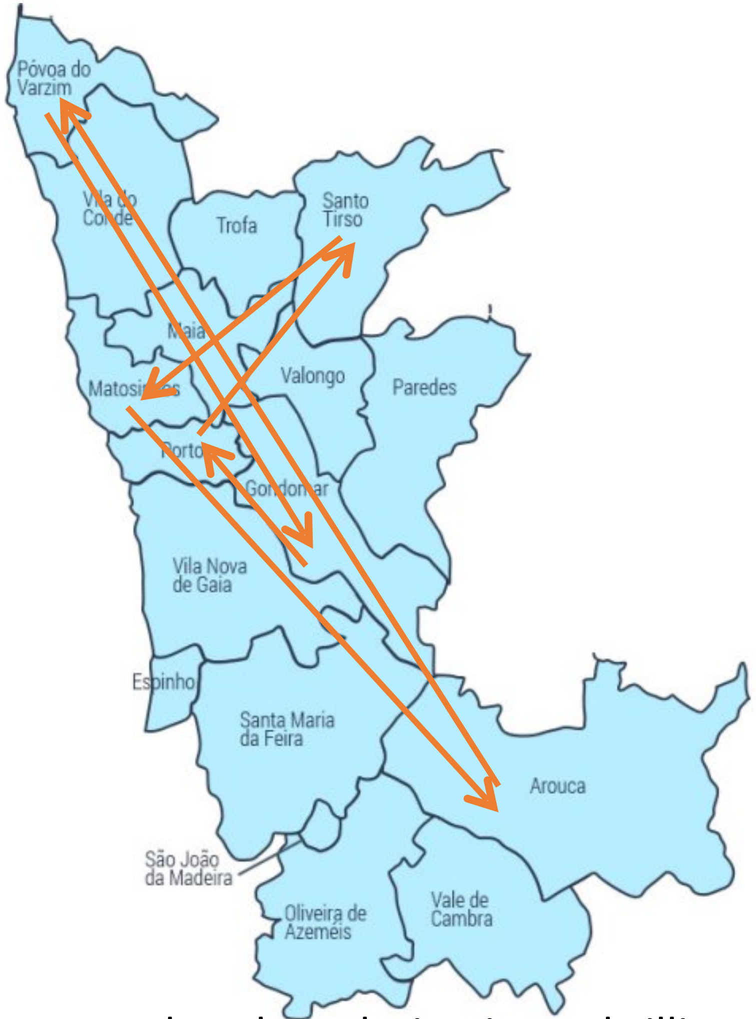
To this first permutation (actually to all of them) we will have to add in the beginning and in the end the main warehouse (start/arrival point - Matosinhos, nr. 5). We will then get the complete sequence [5,1,9,3,8,11,5].

Calculation of the solution time

Sequence: [5,1,9,3,8,11,5]

Using the warehouse identifiers:

```
%idArmazem(<local>,<codigo>)
%warehouseId(<city>,<id>).
idArmazem(‘Arouca’,1).
idArmazem('Espinho',2).
idArmazem('Gondomar',3).
idArmazem('Maia',4).
idArmazem('Matosinhos',5).
idArmazem('Oliveira de Azemeis',6).
idArmazem('Paredes',7).
idArmazem('Porto',8).
idArmazem('Povoa de Varzim',9).
idArmazem('Santa Maria da Feira',10).
idArmazem('Santo Tirso',11).
idArmazem('Sao Joao da Madeira',12).
idArmazem('Trofa',13).
idArmazem('Vale de Cambra',14).
idArmazem('Valongo',15).
idArmazem('Vila do Conde',16).
idArmazem('Vila Nova de Gaia',17).
```



Just looking at the map we can see that the solution is not brilliant…
And it will involve long stretches that may require intermediate battery recharges (which will penalize the total route time)

Calculation of the solution time

The data associated with the travel of the trucks following the sequence of warehouses is represented by the following facts:

```
%dadosCam_t_e_ta(<nome_camiao>,<cidade_origem>,<cidade_destino>,<tempo>,<energia>,<tempo_adicional>).
```

```
%dataTruck_t_e_ta(<truckName>,<start_city>,<end_city>,<time>,<energy>,<aditional_time>).
```

```
dadosCam_t_e_ta(eTruck01,5,1,141,51,24).
dadosCam_t_e_ta(eTruck01,1,9,185,74,53).
dadosCam_t_e_ta(eTruck01,9,3,86,35,0).
dadosCam_t_e_ta(eTruck01,3,8,38,8,0).
dadosCam_t_e_ta(eTruck01,8,11,53,22,0).
dadosCam_t_e_ta(eTruck01,11,5,55,25,0).
```

The time and energy specified in the facts above suppose that the truck has a full load (truck weight+maximum load) of 7500 kg + 4300 kg = 11800 kg (described in the truck facts)

Calculation of the solution time

Truck characteristics:

```
carateristicasCam(eTruck01,7500,4300,80,100,60).
%carateristicasCam(<tipo>,<tara>,<carga_maxima>,<capacidade_bateria>,<autonomia>,<tempo_carga_bat>)
%truckCharacteristics(<type>,<tare>,<maxLoad>,<batteryCapacity>,<autonomy>,<batChargeTime>)
```

The truck leaves Matosinhos (5) with a total load for the 5 deliveries (870 kg in this example), with batteries full (80 kWh) and drives to Arouca (1)

```
dadosCam_t_e_ta(eTruck01,5,1,141,51,24).
%dadosCam_t_e_ta(<nome_camiao>,<cidade_origem>,<cidade_destino>,<tempo>,<energia>,<tempo_adic>).
%dataTruck_t_e_ta(<truckName>,<start_city>,<end_city>,<time>,<energy>,<aditional_time>).
```

Our truck will then start with a total weight of 7500+870 kg = 8370 kg

What will be the time and energy consumption for this travel between cities 5 and 1?

The time (141 minutes) and energy (51 kWh) stated in the fact must be multiplied by 8370/11800 (as seen in the first support document)

$time_{5 \to 1} = 141 * 8370/11800 = 100.014$ minutes

$energy_{5 \to 1} = 51 * 8370/11800 = 36.175$ kWh

Calculation of the solution time

How much energy will be available after the first stage of the route?

$energy_{arrival\ 1} = 80 – 36.175 = 43.825$ kWh

The waiting time for the truck in the warehouse nr 1 (Arouca) will be the **greatest** of the delivery pickup (**unload) time** and the **battery recharge time** (up to 80% - 64 KWh) **if necessary**. In this case the delivery upload time is 10 minutes (last argument in the delivery fact) .

To know if a recharge of the truck battery will be needed, we will have to check which will be the energy consumption during the next stage of the route (1 to 9).

Note that during this 1 to 9 trip the **multiplying factor** will now be (7500+870-200) = 8170 kg because already **200kg were unloaded** at the warehouse 1.

```
dadosCam_t_e_ta(eTruck01,1,9,185,74,53).
```

$time_{1 \rightarrow 9} = 185 * 8170/11800 = 128.089$ minutes

$energy_{1 \rightarrow 9} = 74 * 8170/11800 = 51.234$ kWh

It is clear that we will then need to recharge the truck batteries up to 64 kWh (80% maximum charge) level.

Calculation of the solution time

And how much time this battery recharge will take?

$time_{charge1} = (64 – 43.825) * 60/48 = 25.219$ minutes

As this value is greater than the delivery pickup time it is the one to be considered.

Another issue to be verified is to check whether between leaving a warehouse and arriving to the next the battery level doesn’t go down the 20% limit (16kWh). And precisely that happens between warehouse 1 and the arrival to warehouse 9:

64 - 51.234 = 12.764 kWh (smaller than 16 kWh)

Looking into the fact associated with this stage of the route,

```
dadosCam_t_e_ta(eTruck01,1,9,185,74,53).
```

We will find that the last argument specifies the extra time to recharge the battery in order that it will have on arrival the minimum 20% charge:

$time_{stop\ 1 \rightarrow 9} = 53$ minutes

This time is an estimation and accounts for potential trajectory changes, waiting time and the actual recharge time. Note that this additional charge is applied because in the previous warehouse the battery was topped to the 64 kWh (80%) level.

Calculation of the solution time

As the truck arrives to warehouse 9 with a 20% charge (16kWh) it is clear that the battery must be recharged:

$time_{charge\ 9} = (64 – 16) * 60/48 = 60$ minutes

And as the time used to unload the delivery in 9 is clearly smaller, then the waiting time in this warehouse will be those 60 minutes.

In this warehouse 150 Kg of load will be left, meaning that the total weight of the truck plus load will be 8170 – 150 = 8020 kg.

Let us now calculate the time and energy needed to travel from warehouses 9 to 3.

Looking into the fact associated with this stage of the route:

```
dadosCam_t_e_ta(eTruck01,9,3,86,35,0).
```

We will have:

$time_{9\rightarrow 3} = 86 * 8020/11800 = 58.451$ minutes

$energy_{9\rightarrow 3} = 35 * 8020/11800 = 23.788$ kWh

Calculation of the solution time

How much energy will the truck battery have when arriving to the warehouse 3?

$energy_{arrival\ 3} = 64 – 23.788 = 40.212$ kWh

Let us check whether this energy is enough for the truck to arrive at the next stop (warehouse 8).

The time needed to unload the truck in this warehouse is 10 minutes (last argument in the fact describing this delivery).

To know if there will be a need for a new recharge of the truck batteries we must check the energy consumption in the next stage (from 3 to 8).

Remember that the total weight of the truck plus load will be 100 kg less than before (delivery in 3) being now 7920 kg.

```
dadosCam_t_e_ta(eTruck01,3,8,38,8,0).
```

$time_{3\rightarrow 8} = 38 * 7920/11800 = 25.505$ minutes

$energy_{3\rightarrow 8} = 8 * 7920/11800 = 5.369$ kWh

So the energy when arriving at 3 is more than enough to still drive to the warehouse 8 and the waiting time in 3 will only be the one needed to unload the delivery.

$time_{unload\ 3} = 7$ minutes (last argument of the fact describing the delivery in warehouse 3)

Calculation of the solution time

How much energy will be available in the truck battery after arriving to warehouse 8?

$energy_{arrival\ 8} = 40.212 – 5.369 = 34.842$ kWh

To know if we will have to charge the truck's batteries we need to check the energy consumption during the next stage of the route (from 8 to 11).

In this stage of the route from 8 to 11 the multiplying factor has now become 7920 – 120 = 7800 kg because 120kg have been unloaded in warehouse 8.

```
dadosCam_t_e_ta(eTruck01,8,11,53,22,0).
```

$time_{8\rightarrow 11} = 53 * 7800/11800 = 35.034$ minutes

$energy_{8\rightarrow 11} = 22 * 7800/11800 = 14.542$ kWh

At the arrival at the warehouse 11 the energy in the trucks's battery will be:

$energy_{arrival\ 11} = 34.842 – 14.542 = 20.3$ kWh

That means that there will be no need to charge the battery at warehouse 8 and the waiting time at this warehouse will just be the one needed to unload the truck:

$time_{unload\ 8} = 8$ minutes (last argument of the fact describing the delivery at 8)

Calculation of the solution time

The warehouse 11 is the last destination where to make deliveries, but not the final destination of the truck, which after unloading the last delivery (300 kg) still has to return empty to the Matosinhos (5) warehouse.

In order to know whether or not it will be necessary to charge the batteries one last time, we need to know the energy consumption of the last stretch of the route (from 11 back to 5).

Bear in mind that in this last stage of the route from 11 to 5 the multiplying factor will now be 7800 – 300 = 7500 kg, corresponding to the tare (truck weight).

```
dadosCam_t_e_ta(eTruck01,11,5,55,25,0).
```

$time_{11\rightarrow5}$ = 55 * 7500/11800 = 34.958 minutes

$energy_{11\rightarrow5}$ = 25 * 7500/11800 = 15.890 kWh

When arriving at warehouse 5 the energy in the trucks's battery would be:

$energy_{arrival\ 5}$ = 20.3 – 15.890 = 4.41 kWh

This means that it will be necessary to charge the battery in warehouse 11 to be sure that the battery on arriving at the final destination (5) will have a level above 16 kWh deemed to be the minimum charge level considered safe.

Calculation of the solution time

The general procedure is to charge up to 80% (reaching 64kWh). In this situation the charging time would be:

$time_{charge\ 11} = (64 – 20.3) * 60/48 = 54.625$ minutes

As the time to unload the last delivery is only 20 minutes (last argument of the fact describing the delivery in warehouse 11) then the waiting time would be the charging time because it is greater.

But here we can find a better solution because this way the truck would return to 5 with a remaining battery level of:

64 - 15.890 = 48.11 kWh

But in fact 16kWh would be enough. Therefore the battery charge can be only

16 – 4.41 = 11.59 kWh

where 4,41 kWh is the battery level the truck would have when arriving at 5 without any recharging at 11.

$time_{charge\ 11} = (11.59) * 60/48 = 14.488$ minutes

This value is lower than the 20 minutes unload time. Therefore what determines the warehouse 11 waiting time is the delivery unload time:

$time_{unload\ 11} = 20$ minutes

Calculation of the solution time

Now we can calculate the total time of the sequence [5,1,9,3,8,11,5]:

$time_{5 \to 1}$ = 141 * 8370/11800 = 100.014 minutes

$time_{charge\ 1}$ = (64 – 43.825) * 60/48 = 25.219 minutes

$time_{1 \to 9}$ = 185 * 8170/11800 = 128.089 minutes

$time_{stop\ 1 \to 9}$ = 53 minutes

$time_{charge\ 9}$ = (64 – 16) * 60/48 = 60 minutes

$time_{9 \to 3}$ = 86 * 8020/11800 = 58.451 minutes

$time_{unload\ 3}$ = 7 minutes

$time_{3 \to 8}$ = 38 * 7920/11800 = 25.505 minutes

$time_{unload\ 8}$ = 8 minutes

$time_{8 \to 11}$ = 53 * 7800/11800 = 35.034 minutes

$time_{unload\ 11}$ = 20 minutes

$time_{11 \to 5}$ = 55 * 7500/11800 = 34.958 minutes

Adding these partial times we get **555.27 minutes** (9 hours, 15 minutes and 16,2 seconds)

Calculation of the solution time

How to code this calculation in Prolog?

That's the goal of phase 2, to generate and evaluate all the permutations and to choose the one that corresponds to the smaller time (which will not certainly be the one we illustrated in the previous slides).

The travel times (in red) should already have been taken care of in the phase 1. Now you'll have to deal with the energy assessment that will give us the battery charging times (wine coloured), to be compared with the delivery unload times (being done in parallel, we will choose the smaller) and the eventual need of intermediate battery charges (brown).

We are dealing here with recursive list processing, and for that you will need much less pages of Prolog code than the ones we spent with the explanations of the previous slides!

Now, nobody will prevent the truck from delivering the orders!