CHAPTER 3

A Closer Look at Scoring

Kai Nagel, Benjamin Kickhöfer, Andreas Horni and David Charypar

3.1 Good Plans and Bad Plans, Score and Utility

As outlined in Section 1.4 and by Figures 1.1 and 1.4, MATSim is based on a co-evolutionary algorithm: Each individual agent learns by maintaining multiple plans, which are scored by executing them in the mobsim, selected according to the score and sometimes modified. In somewhat more detail, the iterative process contains the following elements:

mobsim The mobility simulation takes one "selected" plan per agent and executes it in a synthetic reality. This may also be called network loading.

scoring The actual performance of the plan in the synthetic reality is taken to compute each executed plan's score.

replanning consists of several steps:

1. If an agent has more plans than the maximum number of plans (a configuration parameter), then plans are removed according to a (configurable) plan selector (choice set reduction, plans removal).
2. For some agents, a plan is copied, modified and then selected for the next iteration (choice set extension, innovation).
3. All other agents choose between their plans (choice).

An agent's plans in a given iteration may be considered the agent's **choice set** in that iteration. As a result, steps 1 and 2 of replanning modify the choice set, while step 3 implements the actual **choice** between options. Choice is typically based on the score; higher score plans are more likely to be selected. This is discussed in more detail in Chapters 47 and 49. For the time being, note that the three steps of replanning must cooperate for the approach to work: the plans removal step should remove "bad" plans, the innovation step should generate "good" plans, and the choice should, ingeneral,

How to cite this book chapter:

Nagel, K, Kickhöfer, B, Horni, A and Charypar, D. 2016. A Closer Look at Scoring. In: Horni, A, Nagel, K and Axhausen, K W. (eds.) *The Multi-Agent Transport Simulation MATSim*, Pp. 23–34. London: Ubiquity Press. DOI: http://dx.doi.org/10.5334/baw.3.

select good plans. Here, "good" means "able to obtain a high score in the mobsim/scoring". Fortunately, due to its evolutionary concept, the approach is fairly robust: the innovation step does not always have to generate good solutions; it is sufficient if *some* of the solutions are good and lead to a high score.

With this, it is clear that scoring is a central element of MATSim. Only solutions obtaining a high score will be selected by the agent and survive the plans removal step. Thus, the scoring function needs to be "correct" for a given scenario, meaning, more or less, that plans "performing well" obtain a higher score than plans that "do not perform well". Whether a performance is good or not, is decided, in the end, by travelers living in a region: some may prefer a congested car trip, others may prefer a crowded, but affordable, trip by public transit, while others may prefer using the bicycle, even in bad weather.

The typical way to bridge this gap is to use econometric **utility** functions, for example, from random utility models (e.g., Ben-Akiva and Lerman, 1985; Train, 2003) for the score. However, in AI (Artificial Intelligence), utility functions may also be used in a more general way: for example, the score that each individual agent (or the system as a whole) wants to, or should, optimize (Russel and Norvig, 2010). For these reasons, the terms "score" and "utility" are normally interchangeable in the MATSim context. Since we will need the concept of a marginal utility, this chapter will mostly speak of 'utility', since it is a bit unusual to talk about 'marginal score'.

The user can configure numerous parameters to specify the scoring function. When users are ready to extend MATSim in the next part of the book, they will also learn how to plug in their own customized scoring function.

However, because MATSim is based on complete day plans, the application of choice models for parts of day plans only (for example, mode choice) is not straightforward, as detailed in Section 97.4.4. Because of the absence of complete-day utility functions in the literature, MATSim has started with the so-called Charypar-Nagel scoring or utility function (Section 3.2). This scoring function was, at times, modified, extended, or replaced for specific investigations (Section 3.5). Readily applicable estimates for a full-day utility function are not yet available, as discussed in Section 97.4.4.

3.2 The Current Charypar-Nagel Utility Function

3.2.1 Mathematical Form

The first, and still basic, MATSim scoring function was formulated by Charypar and Nagel (2005), loosely based on the *Vickrey* model for road congestion, as described by Vickrey (1969) and Arnott et al. (1993). Originally, this formulation was established for departure time choice. However, all studies performed so far indicate that the MATSim function is also appropriate for modeling further choice dimensions. It is, however, almost certainly not appropriate for activity dropping and activity addition (see Section 3.3).

Basic Function For the basic function, utility of a plan S_{plan} is computed as the sum of all activity utilities $S_{act,q}$ plus the sum of all travel (dis)utilities $S_{trav,mode(q)}$:

$$S_{plan} = \sum_{q=0}^{N-1} S_{act,q} + \sum_{q=0}^{N-1} S_{trav,mode(q)} \tag{3.1}$$

with N as the number of activities. Trip q is the trip that follows activity q. For scoring, the last activity is merged with the first activity to produce an equal number of trips and activities.

Activities The utility of an activity q is calculated as follows (see also Charypar and Nagel, 2005, p.377ff):

$$S_{act,q} = S_{dur,q} + S_{wait,q} + S_{late.ar,q} + S_{early.dp,q} + S_{short.dur,q} \,. \tag{3.2}$$

The individual contributions are defined as follows:

- The expression

$$S_{dur,q} = \beta_{dur} \cdot t_{typ,q} \cdot \ln(t_{dur,q}/t_{0,q}) \tag{3.3}$$

  is the utility of performing activity q, where opening times of activity locations are taken into account. $t_{dur,q}$ is the performed activity duration, β_{dur} is related to the marginal utility of activity duration (or marginal utility of time as a resource, the same for all activities; see Section 3.2.4), and $t_{0,q}$ is the duration when utility starts to be positive.
- The expression

$$S_{wait,q} = \beta_{wait} \cdot t_{wait,q}$$

  denotes waiting time spent, for example, in front of a still-closed store; β_{wait} is the so-called *direct* (see Section 3.2.4) marginal utility of time spent waiting; and $t_{wait,q}$ is the waiting time. We recommend leaving β_{wait} at zero; also see Section 3.2.5.
- The expression

$$_{late.ar,q} = \begin{cases} \beta_{late.ar} \cdot (t_{start,q} - t_{latest.ar,q}) & \text{if } t_{start,q} > t_{latest.ar,q} \\ 0 & \text{else} \end{cases}$$

  specifies the late arrival penalty, where $t_{start,q}$ is the activity starting time q and $t_{latest.ar}$ is the latest possible penalty-free activity starting time (for example, the starting time of the office core hours, or the starting time of an opera or theater performance).
- The expression

$$S_{early.dp} = \begin{cases} \beta_{early.dp} \cdot (t_{end,q} - t_{earliest.dp,q}) & \text{if } t_{end,q} > t_{earliest.dp,q} \\ 0 & \text{else} \end{cases}$$

  defines the penalty for not staying long enough, where $t_{end,q}$ is the activity ending time and $t_{earliest.dp,q}$ is the earliest possible activity end time q. We normally recommend leaving $\beta_{early.dp}$ at zero, except if really good data about this effect is available.
- The expression

$$S_{short.dur,q} = \begin{cases} \beta_{short.dur} \cdot (t_{short.dur,q} - t_{dur,q}) & \text{if } t_{dur,q} < t_{short.dur,q} \\ 0 & \text{else} \end{cases}$$

  is the penalty for a 'too short' activity, where $t_{short.dur}$ is the shortest possible activity duration. We normally recommend leaving $\beta_{short.dur}$ at zero, except if really good data about this effect is available.

The config syntax (config version v2) is approximately

```
<module name="planCalcScore" >
   <param name="performing" value="6.0" />
   <param name="waiting" value="-0.0" />
   <param name="lateArrival" value="-18.0" />
   <param name="earlyDeparture" value="-0.0" />
   <parameterset type="activityParams" >
      <param name="activityType" value="work" />
      <param name="typicalDuration" value="08:00:00" />
```

```
      <param name="openingTime" value="07:00:00" />
      <param name="latestStartTime" value="09:00:00" />
      <param name="closingTime" value="19:00:00" />
      ...
   </parameterset>
   ...
</module>
```

Travel Travel disutility for a leg q is given as

$$S_{trav,q} = \begin{array}{c} C_{mode(q)} + \beta_{trav,mode(q)} \cdot t_{trav,q} + \beta_m \cdot \Delta m_q \\ +(\beta_{d,mode(q)} + \beta_m \cdot \gamma_{d,mode(q)}) \cdot d_{trav,q} + \beta_{transfer} \cdot x_{transfer,q} \end{array} \tag{3.4}$$

where:

- $C_{mode(q)}$ is a mode-specific constant.
- $\beta_{trav,mode(q)}$ is the *direct* (see Section 3.2.4) marginal utility of time spent traveling by mode. Since MATSim uses and scores 24-hour episodes, this is in addition to the marginal utility of time as a resource (again, see Section 3.2.4).
- $t_{trav,q}$ is the travel time between activity locations q and $q+1$.
- β_m is the marginal utility of money (normally positive).
- Δm_q is the change in monetary budget caused by fares, or tolls for the complete leg (normally negative or zero).
- $\beta_{d,mode(q)}$ is the marginal utility of distance (normally negative or zero).
- $\gamma_{d,mode(q)}$ is the mode-specific monetary distance rate (normally negative or zero).
- $d_{trav,q}$ is the distance traveled between activity locations q and $q+1$.
- $\beta_{transfer}$ are public transport transfer penalties (normally negative).
- $x_{transfer,q}$ is a 0/1 variable signaling whether a transfer occurred between the previous and current leg.

The config syntax (config version v2) is approximately

```
<module name="planCalcScore" >
   <param name="marginalUtilityOfMoney" value="1.0" />
   <param name="utilityOfLineSwitch" value="-1.0" />
   <parameterset type="modeParams" >
      <param name="mode" value="car" />
      <param name="constant" value="0.0" />
      <param name="marginalUtilityOfDistance_util_m" value="0.0" />
      <param name="marginalUtilityOfTraveling_util_hr" value="-6.0" />
      <param name="monetaryDistanceRate" value="-0.0002" />
   </parameterset>
   ...
</module>
```

Equation (3.4) is the direct utility contribution of travel; see Section 3.2.4 for the the full indirect utility as well as the relation to the VTTS (Value of Travel Time Savings), and Chapter 51 for a more general discussion.

Note that distance contributes to disutility in two ways. First, it is included in a direct manner via $\beta_{d,mode(q)}$, which is normal for modes involving physical effort, like walking or cycling. Second, distance is also included monetarily via $\beta_m \cdot \gamma_{d,mode(q)}$, which is normal for car or pt mode, where monetary costs increase depending on distance.

3.2.2 *Illustration*

Figure 3.1 illustrates the scoring function. Time runs from left to right. The example shows part of an executed schedule, with home, work, and lunch activities, connected by a car and walk leg.

Activities are scored with concave functions, modeling decreasing returns to spending more time at the same activity. Travel, in contrast, is modeled with downward sloping straight lines, where the slope may differ for different modes of transport and there may be an initial offset (alternative-specific constant). Note the delay between arrival at the workplace and workplace opening time, reflected in no score accumulation during that period. Agents accumulate those scores over a day, reflected in the bottom graph.

When one assumes all other things (particularly travel times) are equal, then agents maximize their score when activity durations are such that all activities have the same slope (= the same marginal utility; red lines). This follows from basic economic theory (cf. Section 51.2), but can also be seen intuitively; if red lines did not all have the same slope, the agent could gain by extending those activities with steeper slope at the expense of others. Clearly, this holds only when all other things remain constant, particularly travel times.

3.2.3 *The "Wrapping Around" of the Utility Function*

The MATSim mobsim typically starts at midnight and runs until all plans have reached their final activity. By itself, the mobsim, is not limited to a day. However, as already stated in Section 3.2.1, the standard scoring function assumes that plans "wrap around" to 24-hour days. Thus, the last activity is merged with the first into one activity. For example, if the first activity ends at 7 am and the last activity starts at 11 pm, then it is assumed that this is the *same* activity, with a duration of eight hours.

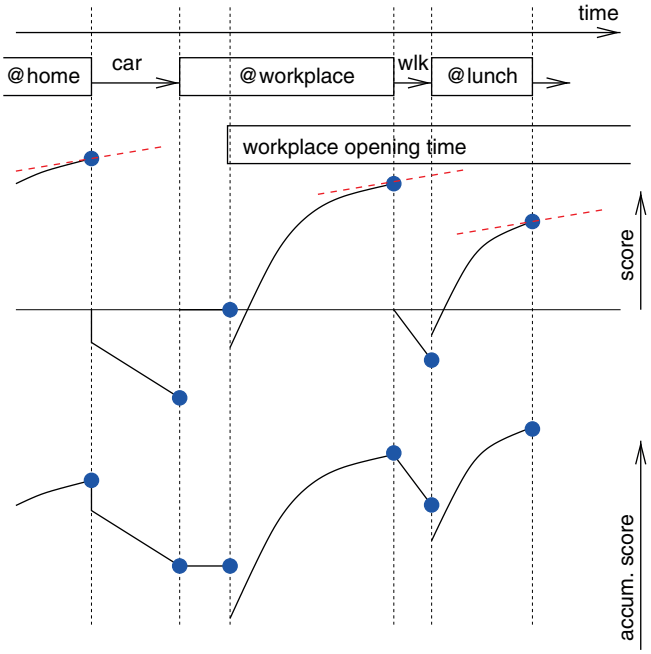

Figure 3.1: Illustration of the scoring function. TOP: Individual contributions of activities and legs. BOTTOM: Score accumulation over a day.

Note that scoring the two activities separately would lead to a different result, because of the nonlinear (logarithmic) form of the utility of performing. For example, $\ln(1) + \ln(7) = \ln(7) \neq \ln(1+7) = \ln(8)$.

3.2.4 MATSim Scoring, Opportunity Cost of Time, and the VTTS

As a result of the wrap-around concept, travel receives, beyond the typically negative direct marginal utility $\beta_{trav,mode}$, an additional implicit penalty from the **marginal utility of time as a resource**: If travel time could be reduced by Δt_{trav}, the person would not only gain from avoiding $\beta_{trav} \cdot \Delta t_{trav}$, but also from additional time for activities (effect of the opportunity cost of time). The **(total) marginal utility of travel time savings** is thus:

$$mUTTS = -\frac{\partial}{\partial t_{trav}} S_{trav} + \frac{\partial}{\partial t_{dur}} S_{dur}.$$

which is

$$mUTTS = -\beta_{trav} + \beta_{dur} \cdot \frac{t_{typ,q}}{t_{dur,q}} \tag{3.5}$$

and at the typical duration of an activity

$$mUTTS\Big|_{t_{dur,q}=t_{typ,q}} = -\beta_{trav} + \beta_{dur},$$

where it can be imagined q is the activity immediately following the trip (cf. Section 51.2). The marginal utility of travel time savings, $mUTTS$, can thus be defined as the indirect effect on the overall time budget, corrected by an offset β_{trav} that denotes how much better, or worse, it is to spend that time traveling, rather than "doing nothing".[1] To differentiate β_{trav} from the indirect effect, it is sometimes called **direct marginal utility** of time spent traveling.

The marginal utility of travel time savings can be transformed to the more common **VTTS (Value of Travel Time Savings)** by dividing it by the marginal utility of money, β_m:

$$VTTS = \frac{mUTTS}{\beta_m} = \frac{-\beta_{trav} + \beta_{dur} \cdot \frac{t_{typ,q}}{t_{dur,q}}}{\beta_m},$$

and at the typical duration of an activity

$$VTTS\Big|_{t_{dur,q}=t_{typ,q}} = \frac{mUTTS}{\beta_m}\Big|_{t_{dur,q}=t_{typ,q}} = \frac{-\beta_{trav} + \beta_{dur}}{\beta_m}$$

This is important for calibration of the utility function.

3.2.5 The Resulting Modeling of Schedule Delay Costs

Arriving Early In the same way as the marginal utility of travel time savings is not only given by $-\beta_{trav}$, but instead by $-\beta_{trav} + \beta_{dur} \cdot \frac{t_{typ,q}}{t_{dur,q}}$, the marginal utility of waiting time savings is given

[1] This is an approximate statement; in the full theory, the reference marginal utility is not given by "doing nothing", but by a Lagrange multiplier related to the constraint that a day has 24 hours; again, cf. Section 51.2.

by $mUWTS = -\beta_{wait} + \beta_{dur} \cdot \frac{t_{typ,q}}{t_{dur,q}}$: Even when the direct marginal utility of waiting, β_{wait}, equals zero, then "doing nothing" still eats into the overall time budget and thus incurs the same opportunity cost of time as traveling does. Intuitively, one can imagine that one must leave the previous activity earlier to have a longer waiting time, thus reducing the score of the previous activity.

Thus, as long as one cannot estimate β_{wait} separately from β_{dur}, we recommend leaving β_{wait} at zero.

Arriving Late Arriving late incurs a marginal utility of β_{late}, typically negative. Here, no additional opportunity cost of time is involved. Intuitively, arriving later implies having left the previous activity later. That is: the current activity is shortened by the same amount that the previous activity was extended, leaving the overall score unaffected (cf. Section 51.2).

Vickrey Parameters As a result, the Vickrey parameters of α (marginal penalty for arriving early), β (marginal penalty for traveling) and γ (marginal penalty for arriving late) (as defined by Arnott et al., 1990) are consistent with the following equations:

$$\begin{aligned} -\beta_{wait} + \beta_{dur} \cdot \frac{t_{typ,q}}{t_{dur,q}} &= \alpha \\ -\beta_{trav} + \beta_{dur} \cdot \frac{t_{typ,q}}{t_{dur,q}} &= \beta \\ -\beta_{late} &= \gamma\,. \end{aligned} \tag{3.6}$$

3.3 Implementation Details

This section summarizes the current implementation of the default MATSim scoring function. The section can be skipped if the reader understands that what has been summarized up to this point is not the full story.

3.3.1 Zero Utility Duration

The duration when an activity's utility is exactly zero is computed by the somewhat cryptic expression

$$t_{0,q} := t_{typ,q} \cdot \exp\left(-\frac{10\,h}{t_{typ,q} \cdot prio}\right), \tag{3.7}$$

where *prio* is a configurable parameter. This is designed so that all activities with the same value of *prio* obtain, at their typical duration, i.e., when $t_{dur,q} = t_{typ,q}$, the same utility value of $10 \cdot \beta_{dur}$, with the idea that this makes them equally likely to be dropped in a time shortage situation (Charypar and Nagel, 2005).[2] However, this does not work as intended, since activities receiving this utility value from a short duration have a larger utility accumulation per time unit than others and are thus dropped later. In consequence, without additional constraints, the "home" activity gets dropped

[2] Starting from Equation (3.3) and inserting Equation (3.7), one obtains

$$\begin{aligned} S_{dur,q}\Big|_{t_{dur,q}=t_{typ,q}} &= \beta_{dur} \cdot t_{typ,q} \cdot \ln\left(\frac{t_{typ,q}}{t_{typ,q} \cdot \exp\left(-10\,h/(t_{typ,q} \cdot prio)\right)}\right) \\ &= \beta_{dur} \cdot t_{typ,q} \cdot \ln\left(\exp\left(10h/(t_{typ,q} \cdot prio)\right)\right) = 10\,h \cdot \beta_{dur}/prio\,, \end{aligned}$$

which is indeed the same for all activities with the same value of *prio*.

first, which is clearly not plausible. See Section 97.4 for a discussion of alternatives. In the meantime, the recommendations are:

- Do not set the `priority` value in the config away from its default value.
- Recognize that the current MATSim default scoring/utility function is not suitable for activity dropping.

3.3.2 Negative Durations

In MATSim, somewhat oddly, it is possible to have activities with negative durations. This can happen because of the "wrap-around" mechanism, where the last activity of a plan is stitched together with the first activity of the plan, and only that merged activity is scored (cf. Section 3.2.3). In this situation, it can happen that an agent arrives at the last activity of the plan at a later 24-hour-time than when the first activity ended. For example, an agent could stay at home until 3 am (end of first activity), then go through her daily plan including a very late party, and return home at 6 am the next morning (Figure 3.2). In this case, the duration of the wrap-around home activity would be *minus* three hours. Originally, a score of zero was assigned to these negative duration activities. However, the adaptive agents quickly found out that they could use this to their advantage, expanding this negative duration without a penalty would lead to more time elsewhere, which the agent could use to accumulate score. For an adaptive algorithm, a penalty like this needs to be defined so that it guides the adaptation back into the feasible region. The penalty must increase with increasing negative duration. It also needs to be more strongly negative than any score value for a positive activity duration. The latter is, however, impossible to achieve with a logarithmic form, which tends to $-\infty$ as $t_{dur,q}$ approaches zero from above. The current approach is to take the slope of the expression $\beta_{dur} \cdot t_{typ,q} \cdot \ln(t_{dur,q}/t_{0,q})$ when it crosses zero, and extend this towards minus infinity (Figure 3.3).

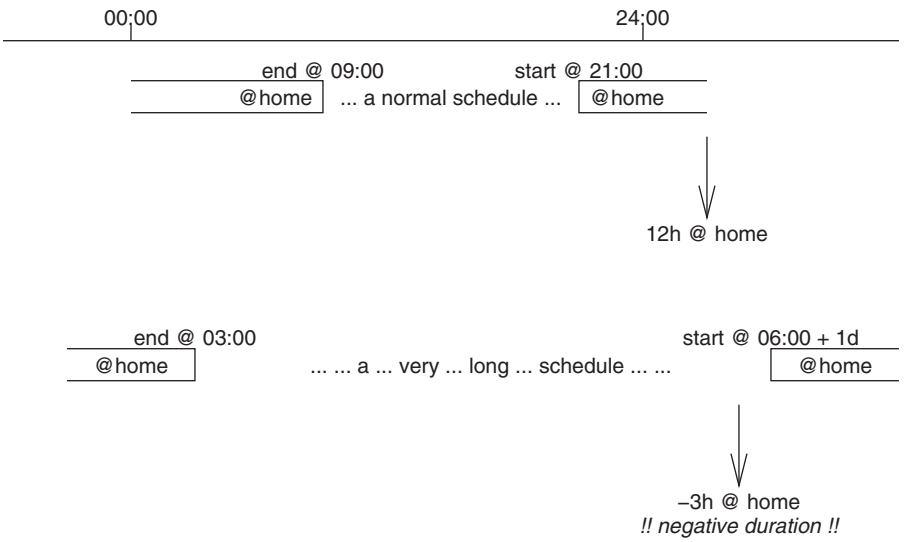

Figure 3.2: Illustration of wrap-around scoring. TOP: Normal situation. BOTTOM: Situation where final activity starts at a later time of day then when the first activity ended, resulting in negative duration.

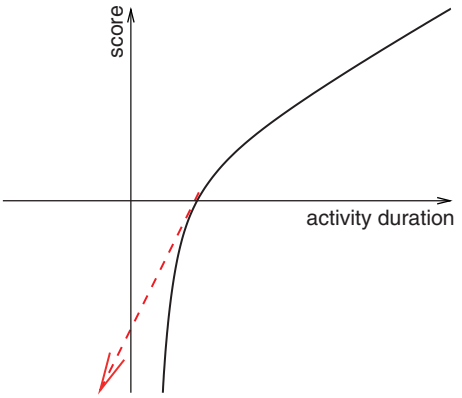

Figure 3.3: Extending the slope when the utility function crosses the zero line to negative durations.

First and Last Activity not the Same Clearly, the wrap-around approach fails if the first and last activity are not the same. The present code does not look at locations, but gives a warning and problematic results if they are of different types.

3.3.3 Score Averaging

The score S that is computed according to the rules given in this chapter is not assigned directly to the plan, rather, it is exponentially smoothed according to

$$S^k = \alpha\, S + (1-\alpha)\, S^{k-1}\,, \tag{3.8}$$

where S^k is the newly memorized score, S^{k-1} is the previously memorized score, S is the score obtained from the plan's execution in the mobsim, and α is a "learning" or "blending" parameter. The default value of α is one; it can be configured by the line

```
<param name="learningRate" value="..." />
```

in the config file.

Non-executed plans just keep their score.

3.3.4 Forcing Scores to Converge

For many situations, both practical and theoretical (see Section 47.3.2.2), it is desirable that each plan's score converges to its expectation value. Equation (3.8) will not achieve that; it just dampens the fluctuations. A well-known approach to force convergence to the expectation value is MSA (Method of Successive Averages):

$$S^m = \frac{1}{m}\, S + \frac{m-1}{m}\, S^{m-1}. \tag{3.9}$$

This resembles Equation (3.8), with two important differences: (1) The fixed blending parameter α is now replaced by a variable $1/m$, and (2) m is not the iteration number but counts how often a plan was executed and thus scored. This is necessary in MATSim since a plan is not executed and scored in every iteration.

This behavior can be switched on by the following config option:

```
<param name="fractionOfIterationsToStartScoreMSA" value="..." />
```

This is plausibly used together with innovation switch off (Section 4.5.3), meaning that MSA operates on a fixed set of plans.

3.4 Typical Scoring Function Parameters and their Calibration

The current MATSim default values are

$$\begin{array}{lrrl} \beta_m & = & 1 & utils/monetaryunit \\ \beta_{dur} & = & 6 & utils/h \\ \beta_{trav,mode(q)} & = & -6 & utils/h \\ \beta_{wait} & = & 0 & utils/h \\ \beta_{short.dur} & = & 0 & utils/h \\ \beta_{late.ar} & = & -18 & utils/h \\ \beta_{early.dp} & = & 0 & utils/h. \end{array} \tag{3.10}$$

They are very loosely based on the Vickrey bottleneck model (e.g., Arnott et al., 1990).

An additional insight is that, in many of the systems that we model, traveling does not seem to be less convenient than "doing nothing". Thus, the *direct* marginal utility of traveling, β_{trav}, is close to zero and sometimes even positive (see, e.g., Redmond and Mokhtarian, 2001; Pawlak et al., 2011). Based on this, a possible approach to calibration is as follows:[3]

1. Set $\beta_m \equiv$ `marginalUtilityOfMoney` to whatever is the prefactor of your monetary term in your mode choice logit model.

   If you do not have a mode choice logit model, set to 1.0. (This is the default.)

   This is normally a positive value (since having more money normally increases utility).

2. Set $\beta_{dur} \equiv$ `performing` to whatever the prefactor of car travel time is in your mode choice mode, while changing that parameter's sign from its typical $-$ to a $+$.

   If you do not have a mode choice logit model, set to $+6.0$. (This is the default.)

   This is normally a positive value (since performing an activity for more time normally increases utility).

3. Set $\beta_{tt,car} \equiv$ `marginalUtilityOfTraveling...` to 0.0.

   It is important to understand this: Even if this value is set to zero, traveling by car will be implicitly punished by the opportunity cost of time: If you are traveling by car, you cannot perform an activity; thus, you are (marginally and approximately) losing β_{dur}. See Section 3.2.4.

4. Set all other marginal utilities of travel time by mode *relative to the car value*.

   For example, if your logit model says something like

   $$\ldots - 6/h \cdot tt_{car} - 7/h \cdot tt_{pt} \ldots,$$

   then

   $$\beta_{dur} = 6\,, \quad \beta_{tt,car} = 0\,, \text{ and } \beta_{tt,pt} = -1\,.$$

   If you do not have a mode choice logit model, set all $\beta_{tt,mode} \equiv$ `marginalUtilityOfTraveling...` values to zero (i.e., same as car).

[3] Different groups have different systems; this one is typical for VSP, although it uses ideas from Michael Balmer.

5. Set distance cost rates `monetaryDistanceRate...` to plausible values, if you have them.

   Note that this needs to be negative: distance consumes money at a certain rate.

6. Use the alternative-specific constants $C_{mode} \equiv$ `constant` to calibrate your modal split.

   (This is, however, not completely simple; one must run iterations and look at the result; especially for modes with small shares, one needs to have innovation switched off early enough near the end of the iterations.)

If you end up having your modal split right, but its distance distribution wrong, you probably need to look at different mode speeds. In our experience, this works better for this than using the $\beta_{tt,mode}$.

Calibrating schedule-based public transport (see Chapter 16) goes beyond what can be provided here.

3.5 Applications and Extensions

The default scoring function has been applied and extended for various purposes. Thus, the historical development is accompanied by various conceptual and technical modifications leading to the current utility function described above. This also means that the reported parameter settings in the literature are an indication, not a direct recommendation.

Important applications for large scenarios are described in Chapter 52.

Special utility functions have been developed for car sharing (see Chapter 22), social contacts and joint trips (see Chapter 28), parking (see Chapter 13), road pricing (see Chapter 15) and destination innovation (see Chapter 27), also describing facility loading scoring and inclusion of random error terms.

Future topics, available on an experimental basis, are: a full-blown utility function estimation (Section 97.4.4), inclusion of agent-specific preferences (Section 97.4.5) and application of alternative utility function forms (Section 97.4).