D5.2 Models of Joint Choice and Joint Use

Project acronym: EUNOIA
Project title: Evolutive User-centric Networks for Intraurban Accessibility
Grant Agreement number: 318367
Funding Scheme: Collaborative project
Project start date / Duration: 01 Oct 2012 / 24 months
Call Topic: FP7-ICT-2011-8
Project web-site: http://www.eunoia-project.eu/
Deliverable: D5.2 Models of Joint Choice and Joint Use
Edition: 1
Date: 04/12/2014
Status: Final
Dissemination Level: Public
Authoring and Approval

<table>
<thead>
<tr>
<th>Prepared by</th>
<th>Thibaut Dubernet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reviewed by</td>
<td>Ricardo Herranz (Deputy Management Coordinator)</td>
</tr>
<tr>
<td>Approved by</td>
<td></td>
</tr>
</tbody>
</table>

Record of Revisions

<table>
<thead>
<tr>
<th>Edition</th>
<th>Date</th>
<th>Description/Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edition 1.Draft 1</td>
<td>24/10/2014</td>
<td>-</td>
</tr>
<tr>
<td>Edition 1.Draft 2</td>
<td>04/12/2014</td>
<td>Minor Revisions</td>
</tr>
<tr>
<td>Edition 1</td>
<td>04/12/2014</td>
<td>Document approved for submission to the EC</td>
</tr>
</tbody>
</table>
Table of Contents

1. INTRODUCTION: SIMULATION OF TRANSPORT SYSTEMS AND INTER-INDIVIDUAL INTERACTIONS .......... 1
   1.1 A SHORT INTRODUCTION TO THE MATSim SOFTWARE FRAMEWORK ......................................................... 1

2. SIMULATING JOINT DECISIONS IN TRANSPORTATION ............................................................................. 3
   1.2 THEORETICAL AND SOFTWARE FRAMEWORK ......................................................................................... 3
      1.2.1 Theoretical Framework .......................................................................................................................... 3
      Joint Decision and Game Theory ................................................................................................................ 4
      Rationality and Altruism in Experiments ...................................................................................................... 4
      A Solution Concept for the Daily Planning Game: the Absence of Improving Coalition ....................... 5
   1.2.2 Software Implementation ...................................................................................................................... 5
      Controller6
   1.3 IMPLEMENTATION .................................................................................................................................. 7
      1.3.1 Selection Operators ............................................................................................................................... 7
      1.3.2 Group Identification for Replanning ..................................................................................................... 7
      1.3.3 Replanning modules ............................................................................................................................ 8
      1.3.4 Scoring Function ................................................................................................................................. 8
      1.3.5 Technical Notes on the Implementation ............................................................................................. 8

3. SIMULATING JOINT USE OF JOINT RESOURCES: THE CASE OF BIKE SHARING .................................. 10
   1.4 DESIGN OF A BIKE SHARING MODEL .................................................................................................... 11
      1.4.1 Supply Side: the Mobility Simulation .................................................................................................. 11
      1.4.2 Demand Side: Agents, Scoring Function and Replanning Operators ............................................. 12
      1.4.3 Technical Notes on the Implementation ............................................................................................. 13

4. REFERENCES ................................................................................................................................................ 15
1. Introduction: Simulation of Transport Systems and Inter-Individual Interactions

Simulation is a powerful tool to better understand and predict the behavior of complex systems. Transportation systems, where a variety of actors with a variety of objectives interact, are an example of a complex system.

This fact has been recognized for more than half a century in transportation science (Wardrop, 1952), and is at the base of the development of agent-based activity-based models, that attempt to predict the state of the transport system by simulating the behavior of a synthetic agent population (see e.g. Bowman, 2009 for an introduction). However, the interactions considered rarely go beyond congestion, whereas individuals in reality interact in a variety of ways:

- road space is not the only joint resource for which individuals compete
- individuals do not only compete, but also cooperate and coordinate with social contacts, by joint travel or joint activities.

This report presents the software development part of the EUNOIA Project, that aims at pushing the limits on those two points. To this end, two streams of development were followed, namely:

- Formulation and implementation of a general framework to represent joint decisions undertaken with social contacts. Specific implementations for joint travel and joint activities, with household members and social contacts, were implemented. Their validation is done as a part of Work Package 6, in the Zurich Case Study.
- Implementation of a simulation for a specific joint resource, of particular interest for the three case studies: public bike sharing schemes.
- This report starts by a general description of the MATSim software framework, that is used as a basis for the two applications presented here, before presenting the details of the two software modules, together with the assumptions behind them. Validation results are part of the report on Work Package 6, as a part of the Zurich Case Study.

1.1 A Short Introduction to the MATSim Software Framework

The proposed simulation tool is built using the MATSim simulation framework. MATSim is an open-source project, usable as a basic standalone simulation software, as an extensible simulation framework, or as a library to build highly specific simulation software (see http://www.matsim.org). It relies on a game-theoretic view of the traffic system, as is frequently done since the seminal work of Wardrop (1952), extending it to the activity-based demand description framework, in which individuals satisfaction depends of contradictory objectives, namely performing activities at different locations while avoiding too much travel. The unification of those two views recognizes that the daily plans performed by individuals lead to usage of joint resources, usually in the form of space (on roads, in buses, at facilities such as shops or restaurants), influencing the satisfaction of other individuals via congestion, or crowding (Balmer, Cetin, Nagel & Raney, 2004; Nagel & Flötteröd, 2009).

This basic formulation leads to an equilibrium problem, which is solved using a co-evolutionary algorithm, embedding this equilibrium solution concept. In short, co-evolutionary algorithms extend the evolutionary algorithm concept by having the solution of the problem split into several components, whose fitness is evaluated.
via \textit{interactions} with other components. Most of the strength (and challenges) of such algorithms is their essentially \textit{game theoretic} nature (Fici, 2004). This makes them particularly well suited for searching for states satisfying a game-theoretic solution concept when the search space is too big for exhaustive search. The interested reader might refer to Popovici, Wiegand & De Jong (2012) for a thorough introduction to this family of algorithms. In the case of daily mobility behavior, the decomposition is natural: individuals in the population are represented by agents, each agent performing an evolutionary algorithm to optimize its daily plan, starting with an initial plan generated externally. The \textit{scores} of daily plans are estimated by letting the agents interact in a \textit{mobility simulation}. This score takes the form proposed by Charypar & Nagel (2005), with a linear disutility of travel time and a logarithmic utility of time passed performing activities. The implementation takes the form of a feedback loop, as illustrated on Figure 1: agents iterate between \textit{replanning} and evaluation, keeping only the plans with the highest score in their memory. A large proportion of the agents simply select an existing plan based on the score, allowing to update the score given the new plans of other agents, as well as reducing the randomness of the simulation to ensure convergence. This process is iterated until the system reaches a steady state, used as an approximation for the equilibrium.

![Figure 1: The MATSim evolutionary process](image)

The software is designed with modularity in mind: changing the \textit{scoring function}, or implementing new \textit{replanning operators} is trivial; adding new behaviors to the \textit{mobility simulation} is reasonably easy, thanks to the agent-based architecture, that separates clearly the behavioral layer from the physical layer. This kind of “framework” usage is for instance sufficient for the bike sharing application presented hereafter.

But more radical modifications of the process can also be undertaken, using the software more as a \textit{library} than a framework. This approach is necessary for the representation of \textit{joint decisions}: a generalized control loop is specified and implemented. Basic elements of MATSim, including mobility simulation, scoring function, replanning operators, or analysis \textit{listeners}, are used as building blocs of the generalized process.
2. Simulating Joint Decisions in Transportation

This work presented in this section aims at developing models of joint choice, with the aim of being used in real-world scenarios.

The main output of this work package is the software itself, used in the Zurich case study; this section highlights the theoretical background, gives implementation details, and links to publications, both released and forthcoming.

The implementation of this simulation platform aims at answering to two important facts identified in the literature: the importance of intra-household coordination on mobility behavior, and the influence of the spatial distribution of social contacts on mobility behavior.


The study of other kinds of social contacts is more limited, most probably due to the difficulty to collect reliable data. Several studies however focused on the influence of social contacts, and joint activities, on travel behavior (Arentze, Kowald & Axhausen, 2012; Carrasco & Habib, 2009; Carrasco, Miller & Wellman, 2008; Deutsch & Goulias, 2013; Habib & Carrasco, 2011; Moore, Carrasco & Tudela, 2013). Such studies reveal significant influences of spatial distribution of social contacts, or of the number of participants in a joint activity, on the traveling behavior of the participants. This is particularly important for leisure purpose trips, which represent an increasing share of the trips performed in western countries (Axhausen, 2005; Schlich, Schönfelder, Hanson & Axhausen, 2004).

Analysis of the behavior and results is part of Work Package 6, as a part of the Zurich case study.

This joint choice concept more specifically relates to two important mobility behaviors: joint travel and joint activities. Those two important behaviors can be understood under the unifying concept of joint decisions: how do individuals, with possibly conflicting objectives but an incentive to collaborate, come to an agreement?

1.2 Theoretical and Software Framework

1.2.1 Theoretical Framework

This section gives a brief overview of the theoretical underpinnings of the simulation framework. More details can be found in Dubernet & Axhausen (2012), Dubernet & Axhausen (2013), Dubernet & Axhausen (2013), or Dubernet & Axhausen (2014).

Game theory, as a theoretical framework to represent competition, has been used in many forms in transportation research. One of the earlier examples, and probably one of the most influential, is the Wardrop equilibrium condition in traffic assignment (Wardrop, 1952), which is simply a Nash equilibrium of a specific
congestion game. This equilibrium notion has then widely spread in transportation research in general, and traffic assignment in particular, and doing an exhaustive review is not the purpose of this paper.

**Joint Decision and Game Theory**

Although the outcome of any game is a decision “joint” in some way (the decision of a player depends on the decisions of the other players), this work uses a more restrictive definition of what is a joint decision.

A joint decision, as we understand it here, is a set of interlinked decisions by several players, requiring the usage of **explicit coordination, or binding agreements**. Including such possibility in a game theoretic framework requires a specific solution concept.

This can be illustrated by a classical game, called the *House Allocation Problem* (Schummer & Vohra, 2007). This game consists of $n$ players and $n$ houses. Moreover, each player has its individual ordering of the houses, from the most preferred to the least preferred, and players prefer being allocated alone to any house rather than in the same house as somebody else. The strategy of a player is the house chosen to live in.

An interesting feature of this game is that any one-to-one allocation of players to houses is a Nash Equilibrium: no player can improve its payoff by *unilaterally* changing its strategy, as it would require choosing an occupied house. This result however contradicts basic intuition about the stability of such an allocation. In this particular case, a more realistic solution concept is the *Absence of Blocking Coalition*: given a one-to-one allocation of houses to players, a blocking coalition is a set of players which could all be better off by re-allocating their houses among themselves.

It is to be noted that both solution concepts correspond to rational agents, *i.e.* agents having a preference ordering over outcomes. What differentiates both solution concepts is the degree of communication which is hypothesized: in a Nash Equilibrium, for a given player, the strategies of the others are taken as given; in an Absence of Blocking Coalition, players have the possibility to “negotiate” a change of strategy with other players, which will be accepted only if all agents in the negotiation are better off after the re-allocations. In this work, we consider that a Nash Equilibrium corresponds to *individual* decisions only, whereas the blocking coalition concept allows what we name *joint* decisions.

**Rationality and Altruism in Experiments**

Experimental data, however, tends to indicate that individuals are not as rational as the classical game theoretic models assume: in laboratory game-playing, players’ behaviors exhibit systematic discrepancies with the game theoretic predictions. In particular, along with the profit-seeking behavior, it seems most individuals are *inequity averse*, in two ways (Fehr & Schmidt, 1999):

1. individuals having the impression to be unfairly treated will attempt to punish the players considered as selfish, even if it implies decreasing their objective (often monetary) payoff;
2. individuals having the impression to get an unfairly better payoff than the others will tend to attempt to increase other player’s objective payoffs, even if it results in a decrease in their own payoff — though this effect is much smaller than the previous one.

Other research, in particular on the special class of games called *social dilemmas* — the games where there exist strategy profiles where all players are better off than at equilibria — also revealed that some *competitive* players actually try to maximize the difference between self and partner, rather than maximizing their own payoff (Kollock, 1998).
Other authors, in particular Rabin (1993) or Falk, Fehr & Fischbacher (2003), model those observations using a “kindness” concept: what matters would not be the material difference of payoffs, as in (Fehr & Schmidt, 1999), but the intention which led to the outcome: individuals tend to be mean to whom they consider being mean to them, and kind to whom they consider being kind to them.

Another way to model seemingly irrational cooperation comes with the introduction of iterations (Kollock, 1998). In this setting, players are assumed to repeat the same game over and over. Each iteration has a decreasing payoff, and the strategy chosen in a given iteration depends on the past plays of the other player. An interesting feature is the fact that in the iterated prisoners dilemma, the “tit for tat” strategy (collaborate until the opponent defects, and then replicate the opponent’s action) is a best response to itself, leading to an equilibrium where the two players eternally cooperate. Intuitively, this kind of model is able to represent some kind of “trust”: players will collaborate to ensure future action of the other player. Leaving with the bounty will increase the payoff for a particular iteration, but the resulting decrease in payoff for the rest of the game is greater than the immediate increase.

This kind of result, though not resulting from what we called a joint decision, is to be kept in mind when modeling joint behavior, as individuals involved in a group decision making process may attempt to achieve a fair agreement — in particular when the members of the group are members of the same household or close friends.

A Solution Concept for the Daily Planning Game: the Absence of Improving Coalition

Given those remarks, a solution concept for the “daily planning game”, including the possibility of binding agreements, is adopted: the Absence of Improving Coalition concept. Given an allocation of daily plans to individuals, an improving coalition is a set of social contacts that could all be better off by simultaneously changing their daily plan. One can think of a group of friends switching from individual dinners at home to a joint dinner in a restaurant.

This concept corresponds to rational agents, because this translates rather easily into algorithmic terms. However, the non-rational effects described above are likely to have a strong influence on joint travel behavior, and should thus be included one way or the other. This is done by transforming them into rational preferences, making agents willing to drive others. More details of the functional form are given below.

1.2.2 Software Implementation

Given this theoretical framework, one needs to design and implement an algorithm to search for allocations of daily plans to individuals, that satisfy this solution concept.

This section presents the EUNOIA implementation, which was built using the MATSim software framework. This implementation consists of two groups of components:

1. A Controller that implements the extension of the MATSim co-evolutionary algorithm, outlined hereafter. It is implemented in a modular fashion, to be easily adapted to the specific need of different simulation scenarios
2. Specific implementations of the modular components, to test the behavior of the approach and apply it to the case studies.
   1. data structures to represent households and generic social networks were implemented, as well as the ways to identify groups for replanning
   2. specific replanning modules to explore the set of possible joint decisions were implemented
The solution algorithm and implementation is a generalization of the MATSim process presented in the introductory session. The following sections outline the different elements implemented in the EUNOIA project.

**Controller**

The MATSim framework provides a Controller to build and configure co-evolutionary algorithms, where agents each optimize their plan given the (evolving) state of the transport system.

Unfortunately, this approach makes choices of agents independent — which of course goes against the simulation of joint decisions. To be able to implement an algorithm searching for states without blocking coalitions, one needs a way to represent the influence of explicit coordination on the utility of a daily plan. This is solved by including joint plans constraints. A joint plan is a set of individual plans executed simultaneously. Different copies of the same individual plan can be part of different joint plans — for instance an agent might go to a given restaurant alone, with members of its household or with a group of friends. The score of the different copies will take into account the influence of the joint plan to which it pertains. Those joint plan constraints are included using heuristic rules, applied after mutation operators are applied, and are classified as strong or weak constraints — weak constraints are considered when selecting plans for execution, but are allowed to be broken when merely selecting plans for mutation. They are then part of the evolution process. In the current application, the heuristic rules consist in joining newly created plans with joint trips (strong) or with leisure activities at the same location at the same time (weak).

To allow handling joint plans, replanning needs to be performed for groups of agents. This is straightforward for households: all agents of a same household are always handled as a single group. For more general social networks, agents are handled with all agents with whom they have a joint plan, plus some social contacts with whom new joint plans can be created. Details are presented in the “Implementation” section.

For each group, two actions are then possible. For most groups, an allocation of existing plans, fulfilling the joint plans constraints, is selected for execution. Based on plan scores, randomized by adding an extreme value distributed error term, an allocation without improving coalitions is searched for by an algorithm inspired by the “Top Trading Cycle” algorithm used for the House Allocation Problem Schummer & Vohra (2007).

For the other groups, a plan allocation is selected and copied. The copied plans then undertake mutation, to make the agents explore new alternative joint plans. What kind of mutation is performed determines which alternative plans will be tried out by the agent. The modules currently implemented are presented in the “Implementation” section.

Agents have a limited memory size, keeping at most 3 plans per joint plan composition, and 10 plans in total. If this limit is exceeded, one should keep the plans which have the highest probability to create improving coalitions, that is, to be preferred to the other plans in the agent’s memory. To this end, a lexicographic ordering is used: the process removes the joint plan which maximizes the number of individual plans which are the worst of the agents’ memories. If several joint plans have the same number of worst plans, the process chooses among them the joint plan which maximizes the number of second worst plans, and so on until the “worst” joint plan is unique. When the overall maximum number of plans in the memory of an agent is reached, the worst individual plan for this agent is removed along with plans of other agents of the same joint plan. Each agents keeps at least one plan not part of a joint plan, as there may otherwise not be any state without blocking coalitions. Agents are parsed in random order, to avoid the emergence of “dictators” over iterations, whose worst plan would always be removed, even if it is the only “bad” plan of a joint plan.
Though those selection operators seem to be in accordance with the chosen solution concept, it is difficult, if not impossible, to prove that the process will actually converge towards the state searched. As noted by Ficici, Melnik & Pollack (2005), when they perform a theoretical analysis of different selection methods in a co-evolutionary context, “Co-evolutionary dynamics are notoriously complex. To focus our attention on selection dynamics, we will use a simple evolutionary game-theoretic framework to eliminate confounding factors such as those related to genetic variation, noisy evaluation, and finite population size”. Those “confounding factors” can however not be eliminated from an actual implementation of a co-evolutionary algorithm, and rigorously proving that a given algorithm actually implements a specific solution concept is very tedious, if not impossible.

With iterations, agents build a choice set of daily plans that becomes better and better given the actions of the other agents. However, the presence of a large portion of agents with plans resulting from random mutation creates noise, not only for the analyst looking at the output of the simulation, but for the agents themselves when they compute the score of their plans. To solve this issue, when the system reaches a stable state, agents stop performing mutation, and only select plans from their memory for 100 iterations, using the absence of improving coalition with randomized scores. This ensures that the selected plans are the result of a behavioral model, rather than the result of random mutation operators.

### 1.2.3 Implementation

The software was designed to be easily extensible and highly modular, to allow both incremental complexification of the various elements, as well as easier maintenance over time.

**Selection Operators**

The first set of elements one can replace are the selector operators, both for co-evaluation and removal. Those elements were presented in the previous section for clarity.

**Group Identification for Replanning**

An important, though purely technical, part of the algorithm, is how agents are grouped together before undertaking replanning. Which agents are replanned together conditions which new joint plans are possible. The requirements are that:

- agents having joint plans together should be replanned together, to allow selecting those joint plans
- any pair of social contacts should have a reasonably high probability to be replanned together
- the replanning groups should not be too big

Two grouping strategies were implemented, depending on the application:

- for households, all members of a household are always replanned as a group. New joint plans can thus always be created between any members of the same household.
- for more generic social networks, another approach has to be found, complying with the requirements above. The method used in the experiments presented here is to first group all agents having joint plans, and then randomly merge such groups, with a probability proportional to the number of social ties between groups.
Replanning modules
The set of possible daily plans that agents will explore is defined by replanning modules. Usual replanning modules modify the plans of agents individually. Additional modules, modifying the plans of several social contacts at the same time, have been implemented, allowing to explore the set of possible social interactions.

Those specific modules are:

- creation of shared rides, by joining a car trip (driver) with a non vehicular mode trip for another agent (the passenger). Currently, the process does not generate pure escorting trips.
- choice of a joint leisure location, by choosing a random location inside of the intersection of approximate space-time prisms for the participating agents

In addition, individual-based modules can be used, such as:

- least cost path routing, given the congested travel times from the last execution step
- departure times random mutation
- mutation of modes at the subtour level, taking into account chaining constraints. A subtour is a sequence of trips starting and ending at the same location, named anchor point. Chaining constraints ensure that vehicles can only be picked-up at the home location.
- switching the sequence of two activities

Scoring Function
Another very important element is the scoring function, that represents the preferences of agents over daily plans. As stated in the introduction, one of the main motivations behind this work is the hypothesis that the willingness to pass time with social contacts has important effects on mobility behavior. This implies that such preferences need to be included in the scoring function.

The following elements were added to the usual MATSim scoring function:

- Preference for joint leisure activities. Three formulations were implemented:
  - Time passed together with social contacts at leisure locations is scored as an activity (with a logarithmic form)
  - Time passed together with social contacts at leisure locations is scored linearly
  - Agents have a preferred number of social contacts to meet at a given leisure activity, and time passed performing this activity with fewer contacts then desired results in a linear penalty.
- Willingness to drive somebody — representing the complex processes described in the theoretical section. This is implemented by a constant in the scoring function for being a driver for somebody.

Technical Notes on the Implementation
The whole framework is available on the matsim public SVN repository: https://svn.code.sf.net/p/matsim/source/

The framework itself, as well as specific elements, is located at: https://svn.code.sf.net/p/matsim/source/playgrounds/trunk/thibautd/src/main/java/playground/thibautd/socnetsim/
Scripts putting the elements of the framework together for specific applications are located in: https://svn.code.sf.net/p/matsim/source/playgrounds/trunk/thibautd/src/main/java/playground/thibautd/socnetsim/run/

Being part of the repository, the framework evolves together with the code base, and is always compatible with the latest state of the MATSim development branch.
3. Simulating Joint Use of Joint Resources: the Case of Bike Sharing

A prominent characteristic of transport systems is that they are composed of a variety of shared and limited resources, that users of the system compete for. Examples of such resources are road space, public transport seats, or, more recently, shared vehicles made available to subscribers of a service.

This last category is of a particular interest for the three EUNOIA case studies: two of the cities, Barcelona and London, already have bike sharing schemes implemented. The third city, Zurich, is in the process of planning such a system.

Game-theoretic simulation frameworks are well suited for studying the effect of the competition for such resources, and have been applied in the past for the cases of road congestion and public transport use. Shared vehicle schemes however, come with new challenges for the modeler:

- the amount of available vehicles is fairly limited, making the availability of a vehicle at a given time, at a given place, much more random than, for instance, the availability of road space.
- the state of the system is not only the result of choices of its users, but also highly depends on reactive strategies of the operator, in particular for vehicle relocation.

Those factors make the representation of such shared vehicle services in an existing game-theoretic simulation platform particularly interesting:

- one can benefit of a software architecture designed to solve this kind of problems, and tested on a variety of test cases
- the particular problems arising from the characteristics of such services help identifying limitations of the algorithms and methodologies applied to more “classical” resources, such as road space. Thus, in the long run, the software framework itself benefits from it, by getting directions of improvement.

The game theoretic framework is a natural way to represent the mutual relations between supply and demand. Those interactions are at the center of lots of the challenges of shared vehicle schemes. One particularly difficult problem is bike redistribution. Indeed, asymmetry in the travel demand and aversion for uphill riding often leads to shortage of bikes in stations with lots of departures and shortage of available slots in stations with lots of arrivals. To address these problems, several approaches can be used. The most common is the usage of relocating trucks, that redistribute bikes during the day. This kind of strategy however comes at a high monetary cost, and might threaten the environmental credibility of bike sharing (Fishman, Washington & Haworth, 2013). Another approach is the usage of incentives to push the system to regulate itself better. Those incentives can typically take the form of station-dependent fares, coupled with an information system. In the case of Paris, for instance, while trips are normally free up to 30 minutes, some stations situated uphill or in the outskirts of the city, labeled “Velib’+”, give right to an additional 45 minutes of free travel time (Nair, Miller-Hooks, Hampshire & Bušić, 2013).

The design of redistribution strategies to allow the provision of a reasonable Level of Service while keeping the costs as low as possible has of course received attention from the research community. Raviv, Tzur & Forma (2013) for instance consider the static redistribution problem, when bikes are only redistributed once during the night. Other Operations Research approaches have included some model of demand, based on historical data, to
design dynamic redistribution strategies, where trucks relocate bicycles throughout the whole day (Contardo, Morency & Rousseau, 2012; Fricker & Gast, 2014; Schuijbroek, Hampshire & van Hoeve, 2013) to satisfy most of the demand. Similar work is also done in the car-sharing field (Barth & Todd, 1999; Kek, Cheu & Chor, 2006). Fricker & Gast (2014) also consider the possibility of “incentives”, which take the form of directives from the operator, such as proposition of an alternative drop-off station, that the user is assumed to follow. Pfromer, Warrington, Schildbach & Morari (2014) considers dynamic price incentives, and estimates their effect by considering customers with a linear value of time. However, none of those approaches contains a model of the reaction of the demand to a change in the quality of the redistribution process. The design of a fully agent based model of bike sharing demand is a first step in the building of a platform to test different relocation strategies, including their effect on demand, and the effects of those changes on demand on the quality of those relocation strategies.

To represent reactions of demand to changes in the supply, explicit behavioral models are needed. A kind of model particularly well suited to the study of bike sharing systems are multi-agent activity-based models such as MATSim. By representing individuals by software agents, they allow to represent individual heterogeneity. More importantly, the possibility to adopt a game theoretic view allows to tackle the challenges linked to competition for limited joint resources.

This report details the simulation framework designed to answer those challenges. The analysis of the performance of this framework is part of Work Package 6, in the case studies for which it was used.

1.3 Design of a Bike Sharing Model

The design of a simulation model of bike sharing demand-supply interaction needs the following components:

1. an accurate representation of the supply side
2. a representation of reactions of the demand to changes in the supply

For the first point, one needs to represent most accurately possible the location of stations, capacity, fares, as well as bike relocation strategies. The second point should be able to represent mode choice, and ideally induced demand.

The second point relies on an accurate formulation of individual preferences. In particular, the difference between uphill and downhill riding should be represented in some way.

Using the activity-based approach, and in particular the MATSim framework, one can nicely represent interdependency between those two layers: agents, representing individuals, have daily plans, including bike sharing trips; they execute them in a simulated world, and their experience depends on the usage of the system by the others. The simulation system searches for an equilibrium state, where, given the choice distribution of the other agents, the choice of a given agent is optimal.

The representation of the two sides is detailed hereafter.

1.3.1 Supply Side: the Mobility Simulation

Representing the supply side is mainly done in the simulation step, from which agents derive the utility of their plans. This step is particularly important in case studies focusing on joint use of joint resources, and even more
for resources as scarce as bikes in a bike sharing system. Hence the system was designed to be highly extensible, to allow iterative complexification, while being able to represent the most salient features of a bike sharing system from the very beginning. The basic design is based on the separation between three entities:

- **Agents**, which navigate in the system, taking and disposing bikes. They include the agents whose daily plan is part of the relaxation process, and are potential users of the bike sharing system. Capabilities of **within-day replanning**, such as heuristic rules to follow in case no bike or no free slot is available, are easily implemented. Those agents can also be **redistribution agents**, which can be as sophisticated as truck drivers reacting to modifications of the state of the system on the fly.
- **Bike sharing stations**, which are containers with a capacity.
- a **Bike Sharing Manager**, which acts as intermediary between agents and stations, checks that capacity constraints are fulfilled, and notifies listeners (e.g. redistribution agents) of changes in the state of the system.

Given this architecture, a variety of bike relocation strategies can be implemented:

- “worst case”, where the system is left to itself
- “best case”, where the system consists of one single “station” accessible from all points where a station is located: all bikes that are not currently in use can be picked up from anywhere, assuming the operator manages to relocate bikes when and where they are needed.
- Schedule-based relocation, where vehicles drive according to a schedule to relocate bikes. This schedule could be evolved with iterations, to be adapted to the observed demand.
- Reactive relocation, where the relocation schedule is computed on-the-fly, reacting to events in the network. This is the kind of strategies for which the simulation approach is the most interesting, but designing such relocation strategies is a project of its own.

### 1.3.2 Demand Side: Agents, Scoring Function and Replanning Operators

For the simulation system to be complete, one needs to extend the capabilities of the agents, roughly speaking extending their choice set of daily plans with bike sharing opportunities. To do so, routing procedures have to be implemented, both for single bike-sharing trips and for bike-sharing stages part of a bigger multi-modal trip. This last part is particularly important when one considers bike sharing as a potential way to solve the “first/last mile” problem. A particularity of routing bike sharing trips is the unpredictability of the availability of bikes at any given station at the time of the trips. Hence, rather than trying to find an “optimal” route, which might contain a station chronically empty, bike sharing routes are produced by choosing a random station in circles centered at the origin and destination points, which are then aggregated into access walk, bike, and egress walk. This randomization helps the agents to try out different possibilities during the co-evaluation phase, allowing the demand to spread over the stations according to their capacity, very much the same way traffic spreads over road paths according to capacity for the case of car traffic.

Given that one of the advantages of bike sharing over a private bike is the possibility to ride bike only for one direction, factors depending on the direction of the trip need to be taken into account. An obvious one is slope: individuals prefer riding downhill than uphill. This fact not only may make somebody prefer bike sharing over private bike, but causes imbalance problems in bike sharing systems, and thus needs to be taken into account. To this end, bike and walk travel speeds are computed taking into account link slopes, age and gender, using the approach from Dobler & Lämmel (2012), itself based on a combination of results on walking and biking speeds.
from the literature. It works by multiplying a reference speed with different factors to take into account age, gender, steepness, as well as a random factor for each person, representing personal variability. No disutility of slope is included yet: though there is strong empirical evidence of the importance of this factor for route and mode choice, the MATSim architecture makes it difficult to get this information into the scoring function.

This of course requires representing adequately chaining constraints in mode choice: if one uses a personal vehicle, the vehicle has to be picked up where it is, and usually brought back home at the end of the day. Bike sharing is not subject to this kind of constraints, contrary to private bike. This has a direct parallel with how mode choice is handled in activity-based travel models, which goes from day-level mode choice (a person is assumed to use the same mode for all trips) to trip-level mode choice. An intermediate granularity is often used: the “subtour” level, which is finer than the rough day-level choice, while keeping it easy to handle chaining constraints. An important feature of one-way vehicle sharing systems is that they allow to break those constraints. The simulation framework thus uses trip-level granularity, but considers chaining constraints: vehicular modes (car and personal bike) have to be used for whole subtours, anchored at a location where the vehicle is available — all vehicles being located at home at the start of the day.

Another advantage of bike sharing over individual bike is its usability as a part of a multi-modal mode chain. Thus, routing for public transport is extended to give the possibility to perform the access or egress trips by bike sharing or walk. This is done the following way:

- for each access mode (walk and bike sharing), all the public transport stations in a circle centered on origin and destination are identified. Maximum radius is 500m for walk, and 5km for bike sharing — the radius being the minimum of this maximum radius and the full trip length.
- routes to all access stops and from all egress stops are computed.
- a standard least-cost path algorithm, based on the public transport schedule, is run between the access and egress stops, to find the optimal route given access and egress disutilities.

The computation of agent and slope dependent walk and bike routes relies on a least-cost path algorithm, making the above approach intractable for scenarios of reasonable size. Fortunately, in the approach from Dobler & Lämmel (2012), which path is optimal for a given origin-destination pair does not depend on the person nor time of day. It makes it possible to cache the paths that the algorithm finds, and only re-compute travel times at future queries. This cache is implemented using soft references, which are a java mechanism allowing to keep as much information in memory as possible, as long as the process does not run out of memory — in which case part of those objects will be cleared. This results in almost all of the queries being answered from the cache rather than the least cost path algorithm, making routing efficient enough for use in a large-scale scenario.

### 1.3.3 Technical Notes on the Implementation

The whole framework is available on the matsim public SVN repository: [https://svn.code.sf.net/p/matsim/source/](https://svn.code.sf.net/p/matsim/source/)

The framework itself, as well as specific elements, is located at: [https://svn.code.sf.net/p/matsim/source/playgrounds/trunk/thibautd/src/main/java/eu/eunoia/project/bikesharing/framework/](https://svn.code.sf.net/p/matsim/source/playgrounds/trunk/thibautd/src/main/java/eu/eunoia/project/bikesharing/framework/)

Scripts putting the elements of the framework together for specific applications are located in: [https://svn.code.sf.net/p/matsim/source/playgrounds/trunk/thibautd/src/main/java/eu/eunoia/project/bikesharing/framework/](https://svn.code.sf.net/p/matsim/source/playgrounds/trunk/thibautd/src/main/java/eu/eunoia/project/bikesharing/framework/)
Being part of the repository, the framework evolves together with the code base, and is always compatible with the latest state of the MATSim development branch.
4. References


