An artificial intelligence framework for compensating transgressions and its application to diet management

An artificial intelligence framework for compensating transgressions and its application to diet management

Luca Anselma[a,*], Alessandro Mazzei[a], Franco De Michieli[b]

[a] *Dipartimento di Informatica, Università di Torino, Corso Svizzera 185, 10149, Torino, Italy*
[b] *Dipartimento di Scienze Mediche, Ospedale San Giovanni Battista, Università di Torino, Torino, Italy*

Abstract

Today, there is considerable interest in personal healthcare. The pervasiveness of technology allows to precisely track human behavior; however, when dealing with the development of an intelligent assistant exploiting data acquired through such technologies, a critical issue has to be taken into account; namely, that of supporting the user in the event of any transgression with respect to the optimal behavior. In this paper we present a reasoning framework based on Simple Temporal Problems that can be applied to a general class of problems,, which we called "cake&carrot problems", to support reasoning in presence of human transgression. The reasoning framework offers a number of facilities to ensure a smart management of possible "wrong behaviors" by a user to reach the goals defined by the problem.

This paper describes the framework by means of the prototypical use case of diet domain. Indeed, following a healthy diet can be a difficult task for both practical and psychological reasons and dietary transgressions are hard to avoid. Therefore, the framework is tolerant to dietary transgressions and adapts the following meals to facilitate users in recovering from such transgressions. Finally, through a simulation involving a real hospital menu, we show that the framework can effectively achieve good results in a realistic scenario.

*Corresponding author
Email addresses: anselma@di.unito.it (Luca Anselma), mazzei@di.unito.it (Alessandro Mazzei), franco.demichieli@unito.it (Franco De Michieli)

Keywords: Artificial intelligence, decision support systems, automatic reasoning, diet management.

1. Introduction

Patient empowerment in healthcare is a challenging goal for medical informatics. The ubiquity of technology allows a pervasive tracking of the human behavior that produce a considerable amount of data. The basic principle of "quantified self", that has recently been adopted to indicate a field of study for the ubiquitous monitoring of human activities, is to use electronic sensors to acquire user data. Quantified self answers the necessity to monitor, to show and sometimes suggest virtuous choices in smart systems designed for personal healthcare. In general, in the personal healthcare domain, users can track some personal data regarding their activities; in fact, it could be useful to exploit such data in order to elicit a virtuous behavior regarding specific goals, e.g., losing weight, following a healthy diet, exercising regularly, increasing water drinking, increasing sleeping time or correctly assuming drugs.

In the personal healthcare domain it is possible to characterize a class of problems with some distinctive features, specifically: (1) a quantitative overall goal to be reached by summing up the contributions of a number of quantitative sub-goals; (2) the presence of constraints that allow for some tolerance in the sub-goals without necessarily precluding the fulfillment of the overall goal; (3) the possibility of non-optimal human behavior, i.e., *transgressions*; (4) the opportunity to compensate earlier transgressions with respect to the fulfillment of the overall goals. For instance, a user may be recommended to score an average number of 10,000 steps per day (sub-goal) for reaching a total of 70,000 steps a week (overall goal). In order to guide a virtuous behavior, a smart system should account that users could not always make all the 10,000 steps per day and, moreover, it should support the users in compensating their transgressions by walking more steps in the following days to reach the overall goal. Another example is related to food: to follow a diet and assume the correct amount of

nutrients over a week (overall goal), it could be useful to compensate a transgression in a meal with the subsequent meals (sub-goals) of the week. We call problems of this class *cake&carrot problems*, to remind that a transgression (e.g., eating cake today) should be compensated by a virtuous behavior (e.g., eating carrots tomorrow).

A smart system coping with cake&carrot problems should also account for a set of challenging issues that, in a realistic scenario, can be associated with such problems, such as (5) a possible indeterminacy in the recording of the numeric value associated to a specific sub-goal fulfillment – for instance, the exact number of steps scored in a walk or the precise amount of eaten food could be unknown; (6) the unavailability of a "perfect" compensating action and thus the necessity to choose among a set of non-optimal possibilities – for instance, a specific destination that allows to perform the required number of steps could not be reached or the perfect meal that counterbalances an excess could not be taken; (7) the presence of possibly contrasting multiple factors to be satisfied at the same time – for instance, if users decide to wake up earlier to go for a walk then they may not get enough sleep, or if they decide for a low-calorie dish they may not reach an adequate protein quote; and (8) the possibility of multiple transgressions, even in following the compensating actions – for instance, in a day following a first transgression, users may be not able to score all the proposed steps, or they may be not willing to eat the proposed compensating meal.

In this paper we describe an artificial intelligence framework – an extension of the Simple Temporal Problem (henceforth, STP) formalism – that is able to deal with cake&carrot problems supporting users wishing to counterbalance a transgressions. For the sake of clarity we chose to illustrate our framework using a dietary scenario as a prototypical example of carrot&cake problems. In this scenario, the reasoning framework acts as a sort of *virtual dietitian* continually supporting the users when dealing with a specific diet.

Following a healthy diet is hard since transgressions are frequent. For example, many people are forced to eat out on a daily basis, relying on a limited

number of dishes to choose from. On the other hand, during holidays meals providing a significantly high calorie content are consumed for several days in a row. Thus, while dietary transgressions are difficult to avoid, when a transgression occurs, it is useful to be aware of its consequences and dynamically adjust the upcoming meals taking into account the impact of such transgressions in order to effectively compensate. Experimental studies show that people may not be able to adjust their diets for psychological reasons too. A well-known phenomenon that shows this peculiar human behavior is called the *Stock-Flow failure*. A very simple metaphor of the dynamics of body weight gain/loss is the bathtub water level, which is determined both by the rate of water flowing in and the rate of water draining out. However, people may encounter some difficulties to adapt their eating behavior in order to take into account this dynamic model. In particular, experimental data show that the necessity to compensate high calorie meals with low calorie meals to regulate the body weight is often disregarded [1].

There are two mains contributions in this paper. The first contribution is methodological and consists in extending STP for supporting cake&carrot problems. More specifically, we have 1) extended STP for supporting also indeterminate constraints and 2) exploited the minimal network computed by the constraint propagation algorithms to offer some reasoning facilities such as, making reference to the dietary use case:

- checking the compatibility of specific meals with specific diets;
- adjusting a diet upon a dieter's actual food intake;
- adjusting a diet upon a dieter's future estimated food intake;
- determining the possible consequences that indulging on specific foods can have on a specific diet;
- simulating the impact of a hypothetical meal on the overall diet;
- choosing the "best" meal among a set of possible meals;

- providing a symbolic evaluation of the adequacy of a meal to the diet;
- supporting imprecision/indeterminacy in food intake.

The second contribution is applicative and consists in 1) the formalization of diets and food intakes in terms of STP to design a system tolerant to dietary transgressions and capable to adjust the rest of a diet, in terms of its macronutrients values, to recover from such transgressions and 2) a quantitative evaluation of the reasoning mechanisms in a controlled realistic hospital scenario, where we considered various types of users and various degrees of transgression disregarding the dietary suggestions provided by the reasoning framework.

The rest of the paper is organized as follows: in Section 2, we introduce some background notions regarding the Simple Temporal Problem framework. In Section 3, we describe the methodological contribution of the paper by proposing several facilities for reasoning in the diet domain. In Section 4, we describe an experimental evaluation of the reasoning module in a realistic context by using the simulation paradigm. Finally, in Section 5 we draw some conclusions and compare our system to related works.

2. Background

In this section we introduce the framework of STP and we point out some limitations that it is necessary to overcome for modeling the class of problems we are addressing.

An *STP constraint* [2] restricts the distance between two time points to be between a lower bound and an upper bound. It has the form $c \leq x - y \leq d$, where x and y are time points and c and d are numbers (both discrete and real numbers are admitted). Also strict inequalities (i.e., $<$) are admitted and $-\infty$ and $+\infty$ can be used for denoting the fact that there is no lower or upper bound, respectively.

An *STP* is a conjunction of STP constraints and it can be represented as a graph whose nodes are the time points and the arcs are labeled with the STP constraint between the points.

A *solution* of an STP is an assignment of a value to each time point such that all STP constraints of the STP are satisfied.

The problem of determining the *consistency* of an STP, i.e., whether there exists a solution, is tractable and it can be solved by detecting whether in the graph there exists a *negative cycle*. All-pairs shortest paths algorithms on weighted graphs such as Floyd-Warshall's algorithms can determine the consistency of an STP and, in case the STP is consistent, they also compute the minimal network [2].

A *minimal network* is an STP equivalent to the original STP (i.e., it has the same solutions) and it makes explicit all the implied STP constraints by representing the strictest constraint between each pair of points, which corresponds to the minimum and maximum distance between each pair of points. A minimal network has the property of being *decomposable*, i.e., any partial assignment of values to time points that is consistent with the STP constraints in the minimal network between those time points can be extended to a solution of the STP [2]. Thus, each value in an STP constraint within a minimal network is present in at least one solution of the STP.

It has been proven that Floyd-Warshall's algorithm, which has a computational cost cubic in the number of time points, is correct and complete on STP [2]. Its temporal computational cost is cubic in the number of the time points.

Limitations of STP for modeling diet dynamics

Usually STP is used to represent time constraints in a static view: first the temporal problem is modeled and then it is checked in terms of satisfiability by propagating the constraints. This modeling methodology does not directly support a dynamic view over the constraints satisfaction problem. In other words, STP does not account for an evolution of the model that is a consequence of the interaction of the system with the external world. This limitation has two important consequences: (1) it does not allow to model system-user interaction and (2) it does not allow for a formal distinction between constraints modeling possibilities on the future and constraints modeling indeterminacy over the past,

which have a different semantics and require a different support.

We will show in Section 3 our extension of the STP model to overcome these limitations in order to dynamically model a diet. In particular, our proposal extends the standard STP framework in many ways:

- it enhances the static STP framework with a number of algorithms to support system-user interaction;
- it proposes the contemporary use of three different STPs to simultaneously consider the three macronutrients;
- it provides a different support for the constraints that represent indeterminacy in the past and the constraints representing variance on the future. In particular, the constraints over the past model indeterminacy/imprecision in the energy intake, whereas the constraints over the future model the different choices that the user can make on the future meal. Thus, it is straightforward to notice that, whatever the user will do in the future, the past constraints should not change. This ontological distinction between past and future constraints is not usually modeled in STP and needs a special treatment in our formalization.

The problem tackled by our extension of STP regarding the indeterminate constraints has some resemblance with the problem of temporal controllability. When dealing with temporal constraints, it is possible to distinguish between constraints under the control of the user or the agent (i.e., controllable constraints) and constraints that depend on the external world, which can be unknown or partially unknown (i.e., non-controllable constraints). These distinction resembles our distinction between STP constraints on future meals, which are controllable, and constraints representing indeterminacy, which are not controllable. Several approaches have been proposed in literature to deal with controllability (see, e.g., [3, 4, 5]), but, in general, their adaptation to the problems we tackled is not trivial. Moreover, in our work we exploit the peculiar topology of cake&carrot problems represented as an STP for reasoning over

constraints representing indeterminacy.

3. Materials and methods

In this section we describe the algorithms underlying the different reasoning facilities that our approach provides. In Section 3 we describe the STP reasoning module. In particular, in Section 3.1 we describe how we model a diet with STP, in Section 3.2 we describe how we exploit STP to represent the food that users consume and its impact on their diets, in Section 3.3 we extend the STP framework to also take into account the eaten food when there is some imprecision/indeterminacy on its quantity and composition. In all three cases we detail how we obtain the minimal network of a diet. In Section 3.4 we discuss some reasoning facilities that can be offered by exploiting the minimal network and we describe the relative algorithms. For the sake of readability we have included in the text only the most comprehensive and novel algorithms; the others can be found in Appendix A.

3.1. Modeling diets in STP

The necessity to encourage people to embrace a healthy diet has been promoted by FAO [6]. Many countries adapted these guidelines in order to blend them into their specific food tradition. In Italy, for example, the Italian Society for Human Nutrition has recently produced a study with recommendations to be used by specialized operators [7]. In particular, it has been proven that an excessive calorie intake – and a possible subsequent obesity – increases the risk of developing chronic disease and decreases life expectancy [8]. In order to restrict the calorie intake, one of the most important factors to be taken into account in a diet is the total energy requirement. Another important factor to consider in a diet is the specific requirement in terms of nutrients and macronutrients such as proteins, carbohydrates and lipids. In particular, the literature on this subject provides systems of Dietary Reference Values (henceforth DRVs) that should be followed for extended periods of time. In the running example,

without loss of generality we refer to the Italian values [7]. Such values have to be customized for the specific patients according to characteristics such as weight, gender, age and lifestyle.

Example. A 40-year-old male who weighs 71.3 kg has an estimated basal metabolic rate of 1690 kcal/day.

The metabolic rate value is then adjusted using a factor related with the specific person's physical activity; for example, if the person has a sedentary lifestyle, a factor of 1.45 is employed and he has a total energy requirement of $1690 \cdot 1.45 = 2450$ kcal/day.

Moreover, [7] prescribes that a population reference intake for each macronutrient is the DRV that covers the physiological needs of most of the healthy or apparently healthy population (97.5% of people). In [7] the DRVs for a male adult are computed by considering 0.9 grams of proteins a day for each kilogram of the patient's weight, the lipids must be between 20% and 35% of the total energy requirement, the carbohydrates cover the remaining energy requirement. In the example, following [7], the daily diet is recommended to be provided with 260 kcal/day of proteins, 735 kcal/day of lipids and 1455 kcal/day of carbohydrates. Our framework does not depend on the particular formula used to compute the total energy requirement or the macronutrients requirements, and it is possible to employ other formulas or the dietitian can even empirically estimate the value for specific patients.

We represent the DRVs as STPs; more precisely, we use an STP constraint to represent – instead of temporal distance between time points – the minimum and maximum admissible DRV for a macronutrient.

In our system and in the experiments in Section 4 we adopt three separate STPs, one for each macronutrient, *CarbohydratesDietSTP*, *ProteinsDietSTP* and *LipidsDietSTP*, and we consider separately the intake and the requirement of each macronutrient. For the sake of readability, in this section, instead of considering three macronutrients and three STPs, we focus the discussion only on the total energy requirement represented as the STP *DietSTP*; however, the same discussion should be applied for each STP modeling a macronutrient.

Indeed, the results that we prove in this section for one single STP are still valid in the case of multiple STPs since the correctness and compatibility of the whole system depends on the correctness and compatibility of each single STP independently evaluated. Note that the idea of using distinct STPs for modeling different macronutrients can be used for cake&carrot problems to simultaneously account for different activities. For instance, sleeping and running activities, which are competing in the daily time scheduling, could be accounted for by using two different STPs in order to indirectly verify their compatibility.

Moreover, without loss of generality, we focus the discussion on a diet over a week, from Sunday to Saturday.

For representing the total energy requirements of the example in Section 3.1, the user's daily energy requirement of 2450 kcal/day is represented in *DietSTP* by the STP constraint $2450 \leq day_E - day_S \leq 2450$, where day_E and day_S stand for the start and the end of the day, and the recommendation to eat a lunch of minimum 500 kcal and maximum 600 kcal is represented by the STP constraint $500 \leq lunch_E - lunch_S \leq 600$, where $lunch_E$ and $lunch_S$ represent the end and the start of the lunch.

We exploit the STP framework to allow users to make small deviations from the ideal goal of sticking to their energy requirements and to know in advance what are the consequences of such deviations on the rest of the diet. Thus, we admit a tolerance in adhering to the diet constraints. Such a tolerance generates loose constraints over the shortest periods (i.e., days and meals) and strict constraints over the longest periods (i.e., weeks), thus allowing episodes of transgression in the short term that however do not prevent a user from reaching the diet goal. Let σ_w, σ_d and σ_m be the tolerances for the week, the days and the meals respectively, then $\sigma_w \leq \sigma_d \leq \sigma_m$. For example, the recommended energy requirement of 2450 kcal/day, considered over a week and considering a tolerance σ_w, results in a constraint such as $2450 \cdot 7 \cdot (1 - \sigma_w) \leq week_E - week_S \leq 2450 \cdot 7 \cdot (1 + \sigma_w)$ and for the single days we allow the users to have a tolerance σ_d, thus resulting in the constraints $2450 \cdot (1 - \sigma_d) \leq Sunday_E - Sunday_S \leq 2450 \cdot (1 + \sigma_d)$, $\ldots$, $2450 \cdot (1 - \sigma_d) \leq Saturday_E - Saturday_S \leq 2450 \cdot (1 + \sigma_d)$

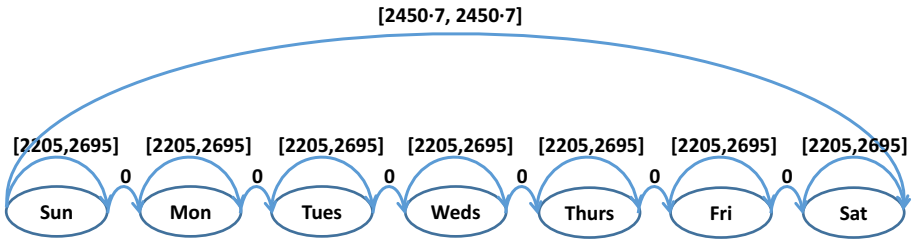

Figure 1: Example of STP representing the diet of a week (with $\sigma_w = 0$ and $\sigma_d = 10\%$). For space constraints the constraints for the meals are not represented and the constraint $[0, 0]$ is represented as 0.

(see Fig. 1).

The choice of the values for σ is influenced by two classes of factors, i.e., general requirements of the framework and specific domain factors. Two general requirements arise from considerations about the usefulness of the framework and are valid for all problems in the cake&carrot class. The first general requirement is that not all the tolerance values can be zero. Indeed, zero tolerance would not allow transgressions and the framework would not be useful. The second general requirement is that tolerances over the longer periods need to be stricter than tolerances over the shorter periods, e.g., $\sigma_w < \sigma_d < \sigma_m$. Indeed, if a shorter period has a tolerance smaller than a longer period, the longer period will be dominated by such smaller tolerance (e.g., a tolerance of 10% for each meal results in a tolerance of at most 10% for the whole week) and its usefulness would be undermined. Specific domain factors derive from the domain knowledge and can provide further elements to properly set up the values of σ. These factors can be obtained from evidences in the literature and from human expertise. In the diet domain, for example, it is possible to take into account the *tolerable upper intake level* for a macronutrient [7], which is the maximum value that an adult can assume without adverse effects. In 17 studies summarised by Bingham [9], mean within-individual coefficient variation for various nutrients in adults were, when measured daily, Energy 23%, Carbohydrate 23%, Protein 27% and,

when measured weekly, Energy 11%, Carbohydrate 11%, Protein 13%. Human expertise can further refine the tolerance values to adjust them on the basis of specific patients' requirements and dietitians' personal empirical experience.

For single meals we consider that the energy assumption for the day is split among the meals; for example, assigning 20% for breakfast, 40% for lunch and 40% for dinner and admitting a tolerance of σ_m, we obtain the STP constraints $2450 \cdot 20\% \cdot (1-\sigma_m) \leq Sunday\_breakfast_E - Sunday\_breakfast_S \leq 2450 \cdot 20\% \cdot (1+\sigma_m)$, $2450 \cdot 40\% \cdot (1-\sigma_m) \leq Sunday\_lunch_E - Sunday\_lunch_S \leq 2450 \cdot 40\% \cdot (1+\sigma_m)$, ..., $2450 \cdot 40\% \cdot (1-\sigma_m) \leq Saturday\_dinner_E - Saturday\_dinner_S \leq 2450 \cdot 40\% \cdot (1+\sigma_m)$.

In order to build a sound STP, it is also necessary to add STP constraints such the ones that impose that a day ends when the subsequent day starts, that a week starts when the first day of the week starts, that a week ends when the last day of the week ends and the analogous constraints for the meals.

A *compatible* meal is a meal that is consistent with all the STP constraints in *DietSTP*. By propagating the constraints of *DietSTP* we obtain its minimal network, *DietMN* (Algorithm A.1 in Appendix A), from which it is possible to derive all the possible compatible meals. Since a diet is represented as an STP, for checking the consistency of a diet is sufficient to check the consistency of the STP with the Floyd-Warshall's algorithm, which also returns the minimal network. The following property trivially derives from the correctness of Floyd-Warshall's algorithm for STP.

Property 3.1. *Algorithm A.1 is correct.*

We can state the following property, which trivially follows from the properties of STP.

Property 3.2. *The solutions of the STP DietSTP – and of the minimal network DietMN – adhere to the dietary recommendations.*

In particular, since all solutions of *DietSTP* must satisfy all the STP constraints, the introduced local deviations do not deter from adhering to the dietary requirements.

3.2. STP reasoning over a diet and eaten food

In the previous section we only took into account the "ideal" and "abstract" diet, without considering the fact that users, by eating food, limit the range of choices available to them in their subsequent meals. Thus, in order to usefully employ our system and exploit the facilities described below, it is necessary to "integrate" the dietary recommendations with the information about the eaten meals. In this section we consider the simplest case where we assume that there is complete knowledge over the eaten food and that, obviously, the food that will be eaten in the future meals is unknown.

The values of the macronutrients of the food actually eaten can be integrated with the values of the macronutrients provided by the diet (see Algorithm A.3) by provisionally substituting the STP constraint related to the considered meal with the actual value of the eaten food. Thus, we obtain a new STP that we call $integratedDietSTP$. Such an STP contains "past constraints", which correspond to the meals taken, and "future constraints", which correspond to the meals that have to be eaten. The constraint propagation on $integratedDietSTP$ results in a new minimal network $integratedDietMN$.

For example, let us suppose that on Sunday, Monday and Tuesday the user had an actual intake of 2690 kcal for each day, and on Wednesday an intake which has been imprecisely measured to be between 2350 and 2400 kcal. This corresponds to adding to the STP the new constraints $2690 \leq Sunday_E - Sunday_S \leq 2690$, ..., $2690 \leq Tuesday_E - Tuesday_S \leq 2690$. Then, propagating the constraints of the new STP, we discover (see Fig. 2) that (i) the STP is consistent and thus the intake is compatible with the diet, (ii) on each remaining day of the week the user must assume a minimum of 2205 kcal and a maximum of 2405 kcal and (iii) overall, in the remaining four days of the week, the user must assume $2270 \cdot 4$ kcal.

For reasons similar to those exposed in the previous section it is possible to state that:

Property 3.3. *Algorithm A.3 is correct.*

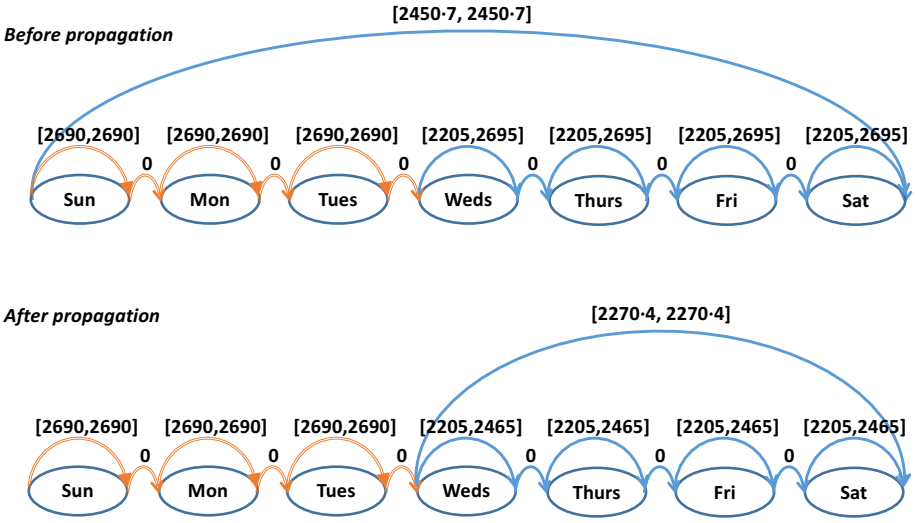

Figure 2: Example of STP for a diet over a week where the user has eaten 2690 kcal on Sunday, Monday and Tuesday. The top part represents the STP before the constraint propagation (it corresponds to the bottom STP of Figure 1 with the new constraints for Sunday, Monday and Tuesday) and the bottom part represents the minimal network obtained by the constraint propagation. and the bottom part represents the minimal network obtained by the constraint propagation. The constraints on eaten food are represented with an orange double line. For the sake of clarity only the main constraints are represented.

3.3. Supporting imprecision/indeterminacy in the eaten food

In this section we extend the work of the previous section in order to support also some imprecision/indeterminacy in the eaten food. Indeed, often it is not possible to precisely know the size of the portions or the composition of the dishes [10]. The actual amount of food in a portion could vary and, furthermore, a user may not eat a whole portion. Supporting such a feature poses new problems and requires an extension of the basic STP framework outlined in the previous section.

At first, we could conjecture that this extension can be addressed in a simple way: in the process described in Section 3.2 we simply substitute the precise value of the eaten food (i.e., $EnergyIntake$) with an interval $[EnergyIntake_{min}$,

$EnergyIntake_{max}$] that represents the imprecision/indeterminacy: the actual real value of the energy intake – which is unknown – is included in the interval [$EnergyIntake_{min}$, $EnergyIntake_{max}$]. Such a choice would be homogeneously represented by an STP constraint, thus it would fit seamlessly in the STP framework.

Unfortunately, this simple approach cannot be adopted because it disregards an important aspect that did not emerge in the previous sections. Indeed, as discussed in Section 2, an STP is consistent if there exists an assignment of a value to each point of the STP such that all the STP constraints are satisfied. Thus, *an STP constraint* assumes that the user can freely choose a value consistent with the constraint. However, the *indeterminate constraints* are of a different nature. In fact, the user is not free to choose a specific value for the energy intake of the meal: the value has been already fixed (indeed the meal has been already eaten) but its actual value is unknown. If we want to be sure to adhere to the dietary constraints, we need to make sure that they are satisfied whatever is the actual value of the indeterminate constraints, and not only for a suitable choice of such value. Thus, in this case a diet is consistent if, whatever is the value assumed in the indeterminate constraints, the STP is consistent.

For example, let us suppose, continuing the previous example, that, after the assumption of 2690 kcal on Sunday, Monday and Tuesday, on Wednesday the user had an intake which has been imprecisely measured to be between 2300 and 2600 kcal. This corresponds to adding to the previous STP the indeterminate constraint $2300 \leq Wednesday_E - Wednesday_S \leq 2600$ (see Fig. 3). Then, propagating the constraints of the new STP by using the standard constraint propagation, we would discover (see Fig. 3) that (i) the STP is consistent and thus the intake is compatible with the diet, (ii) on each remaining day of the week the user has to assume a minimum of 2205 kcal and a maximum of 2370 kcal and (iii) overall, in the remaining three days of the week, the user has to assume between $2205 \cdot 3$ and $2260 \cdot 3$ kcal. However, the indeterminate constraint has been changed by the constraint propagation and, specifically, it cannot assume

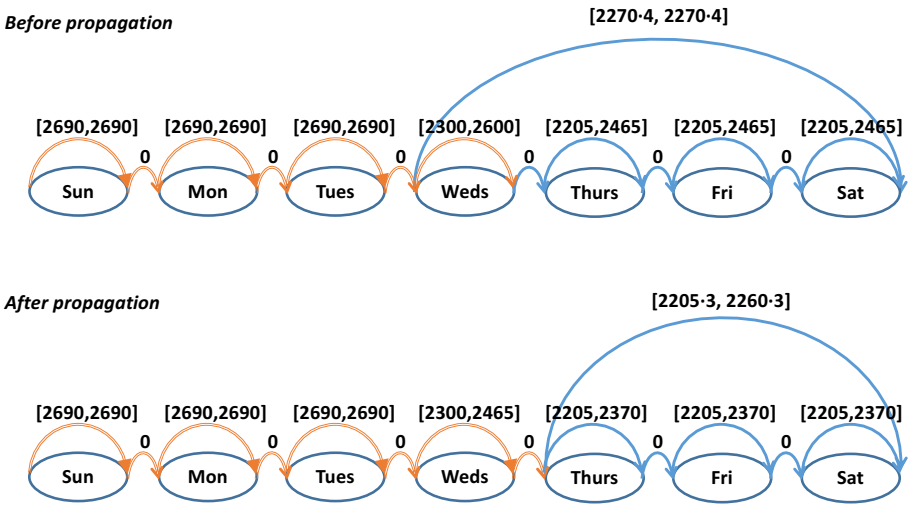

Figure 3: Example of minimal network for a diet over a week where the user has eaten 2690 kcal on Sunday, Monday and Tuesday and between 2300 and 2600 on Wednesday. The top part represents the STP before the constraint propagation (it corresponds to the bottom STP of Figure 2 with a new constraint for Wednesday) and the bottom part represents the minimal network obtained by the constraint propagation. Indeterminate constraints are represented with an orange double line. For the sake of clarity only the main constraints are represented.

anymore values between 2465 kcal and 2600 kcal. If the actual unknown intake on Wednesday is of 2600 kcal, the STP will be inconsistent and the user will not be able to adhere to the diet. Thus, the simple application of the STP constraint propagation in case of indeterminate constraints can give misleading results.

Since, for the reasons discussed above, for checking the consistency of an STP with indeterminate constraints it is not possible to simply apply the Floyd-Warshall's algorithm, we have to extend the Floyd-Warshall's algorithm to support also indeterminate constraints. Algorithm 1 accepts an STP $DietSTP$ and a list of *indeterminate constraints*. Each indeterminate constraint specifies the meal, the day of the meal, and the minimum and maximum values for the energy intake. Algorithm 1 returns the minimal network if the diet is consistent with

the food expressed with indeterminate constraints. On the other hand, if the diet is not consistent, Algorithm 1 distinguishes two separate cases. If the STP is not consistent in the sense that it is not possible to have compatible meals even by restricting the indeterminate constraints, it returns *Inconsistent*. If the STP is consistent for some suitable choice of the indeterminate constraints but not for all the choices, then it returns $Warning$ so that the user can be informed that it is possible that the intakes are not consistent with the diet.

Algorithm 1 can be split in four parts. In the first part (lines 3–6) it does not differ significantly from Algorithm A.3 reported in Appendix A, i.e., the constraints regarding the food are added to the diet STP and the constraints are propagated. In the second part (line 8) we check whether the propagation has changed the indeterminate constraints getting rid of some value (this implies that there is no solution of the STP which contemplates such values) and in this case we return a warning. In the third and fourth parts (lines 10–13 and 15–18) we test whether the STP is consistent when the indeterminate constraints take their minimum and maximum values. Finally, the minimal network computed in line 5 is returned.

We can state the following property:

Property 3.4. *Algorithm 1 is correct.*

Proof (sketch). Let us assume that the STP is consistent, i.e., there is at least an assignment of a value to each point such that all the constraints are satisfied (the other case is ruled out in line 6 of the algorithm). After the propagation, the check in line 8 assures that no value in the indeterminate constraints has been ruled out by some constraint: indeed, a value is in a constraint in the minimal network if and only if it is admitted in a solution of the STP. It remains to check that a solution exists for every combination of such values. This is done by testing the two extreme cases (lines 10–13 and 15–18), i.e., the ones where each indeterminate constraint has the minimal or the maximal values. An STP constraint that is satisfied in these two worst cases is satisfied also in the other cases.

Algorithm 1 Computing the minimal network of a diet with respect to the meals taken (with indeterminacy)

1: **function** MinimalNetworkOfADietWithFoodAndImprecision($DietSTP$, $\langle Day_1, MealType_1, EnergyIntake_{1,min}, EnergyIntake_{1,max}\rangle$, ..., $\langle Day_n, MealType_n, EnergyIntake_{n,min}, EnergyIntake_{n,max}\rangle$)
2:
3: let $integratedDietSTP$ be a provisional copy of $DietSTP$
4: add to $integratedDietSTP$ the indeterminate constraints
5: $integratedDietMN \leftarrow$ FloydWarshall($integratedDietSTP$)
6: **if** $integratedDietMN$ has a negative cycle **then return** $Inconsistent$
7:
8: **if** in $integratedDietMN$ an indeterminate constraint has been changed by the propagation **then return** $Warning$
9:
10: let $MinMN$ be a provisional copy of $integratedDietMN$
11: add to $MinMN$ the indeterminate constraints set to their minimal values (i.e., $\forall i = 1 \dots n\ EnergyIntake_{i,min} \leq E - S \leq EnergyIntake_{i,min}$)
12: $MinMN \leftarrow$ FloydWarshall($MinMN$)
13: **if** $MinMN$ has a negative cycle **then return** $Warning$
14:
15: let $MaxMN$ be a provisional copy of $integratedDietMN$
16: add to $MaxMN$ the indeterminate constraints set to their maximal values (i.e., $\forall i = 1 \dots n\ EnergyIntake_{i,max} \leq E - S \leq EnergyIntake_{i,max}$)
17: $MaxMN \leftarrow$ FloydWarshall($MaxMN$)
18: **if** $MaxMN$ has a negative cycle **then return** $Warning$
19:
20: **return** $integratedDietMN$

Finally, it is worth noting that constraints on the future can, in some special cases, be treated as expressing a different semantics. Suppose that it will be Christmas in few days and that the users know in advance that they will overeat for the Christmas lunch. They could set in advance a higher (estimated) caloric intake for the Christmas lunch so that they could exploit the STP also in the preceding days for compensating the expected transgression. In this case, this future constraint has to be treated as an indeterminate constraint in the Algorithm 1. In other words, a future constraint can be used in the framework to express possible intervals as well as indeterminacy in caloric values.

3.4. Reasoning facilities

In this section we show how we exploit the minimal network obtained by the algorithms described above to offer some reasoning facilities. In particular, we wish to support users into taking advantage of the information regarding the actual meals they consume.

Our system, by exploiting the STP framework, offers several facilities:

1. Consistency check of a diet;
2. Compatibility check of a single meal;
3. Compatibility check of several meals;
4. List of the dietary constraints;
5. What-if analysis;
6. Choice of the best meal;
7. Evaluation of a meal.

Consistency check of a diet. Such a facility is directly provided by Algorithms A.1, A.3 and 1. Indeed, the Floyd-Warshall's algorithm determines the consistency of an STP by detecting a negative cycle in the minimal network. Moreover, Algorithm 1 extends such a mechanism by also detecting inconsistencies related to the indeterminate constraints.

Compatibility check of a single meal. Such a facility is provided by Algorithm A.2 in Appendix A by exploiting the decomposability of the minimal

network. Indeed, since each constraint of the minimal network contains values that take part at least to a solution and the minimal network has the same solutions of the original STP, if a meal is not included in its related STP constraint in the minimal network, then it is not consistent with the STP and with the diet.

Property 3.5. *Algorithm A.2 is correct.*

Compatibility check of several meals. Such a facility cannot be provided simply by repeating Algorithm A.2 once for each meal because in this way we would make sure that the meals are individually consistent with the diet but we would not be sure that they are conjunctively consistent with the diet. In other words, we would possibly conclude that the each meal is present in a solution of the STP but not necessarily they would be present in the same solution. For these reasons we have to proceed as in Algorithm 2.

Algorithm 2 Compatibility check of several meals with respect to a diet

function CompatibilityCheckOfSeveralMeals($MinimalNetwork$, $\langle Day_1, MealType_1, EnergyIntake_1\rangle, \ldots, \langle Day_n, MealType_n, EnergyIntake_n\rangle$)
  for each $(MealType_i, Day_i)$ **do**
    let $a \leq E - S \leq b$ the STP constraint in $MinimalNetwork$ over the starting and ending points of $MealType_i$ in Day_i
    if $a \leq EnergyIntake_i \leq b$ **then**
      add provisionally to $MinimalNetwork$ the constraint $EnergyIntake_i \leq E - S \leq EnergyIntake_i$
    else
      return $False$
  $MinimalNetwork' \leftarrow$ FloydWarshall($MinimalNetwork$)
  if $MinimalNetwork'$ has a negative cycle **then**
    return $False$
  else
    return $True$

For determining whether several meals are consistent with the diet at the same time, first we check whether the single meals are inconsistent. In this case, obviously neither the meals collectively can be consistent. Then, if all meals are separately consistent, we provisionally add them to the minimal network and

propagate the constraints. If the resulting STP is consistent, we can be sure that there exists a solution of $DietSTP$ that includes all the values for the provided meals; if it is not consistent, the meals are separately consistent but not conjunctively consistent. We can state that:

Property 3.6. *Algorithm 2 is correct.*

This property directly derives from the correctness of Floyd-Warshall's algorithm for STP.

List of the dietary constraints. In such a facility provided by Algorithm A.5 in Appendix A we simply traverse the minimal network and, for each day and for each meal of the day we report, possibly graphically, the minimum and maximum admitted value of the related STP constraint.

What-if analysis. With such a facility provided by Algorithm A.4 in Appendix A we enable users to become aware of the consequences on their diets of eating specific meals, thus allowing them to use such information to make informed decisions about their current or future meals.

In the algorithm we assume that all the energy intakes are compatible with the diet. The algorithm adds the STP constraints stating the amount of the energy intake of each meal to a provisional copy of $DietMN$, then it propagates the constraints to determine the new minimal network and, exploiting the previous facility, it shows to the user the STP constraints related to the rest of the meals of the week.

Evaluation of a meal. Such a facility, provided by Algorithm A.7 in Appendix A, allows to symbolically evaluate a meal. While Algorithm A.2 can be employed to check the consistency of a meal and Algorithm A.6 can be employed to automatically choose the "best" meal, such a facility can be used to provide the users with a high-level and more intuitive evaluation of a proposed meal with a richer answer than a simple consistent/not consistent one. This facility is intended to transform a numerical result, such as the one obtained through STP, into a symbolic form. Such a form can be possibly coupled with

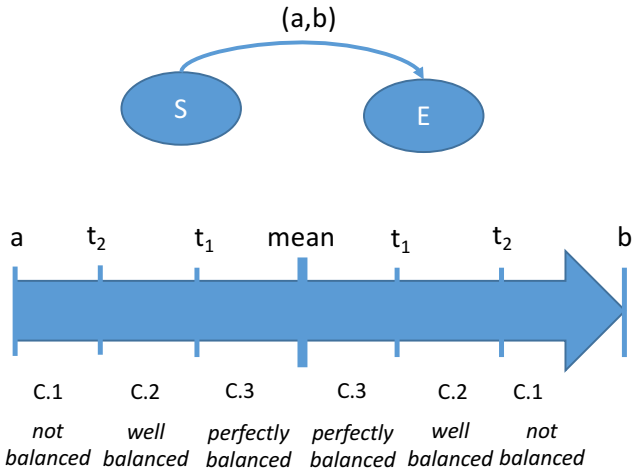

Figure 4: Classification of a consistent value of a meal's energy supply given the minimum and maximum value of an STP constraint in a minimal network.

a graphical result or a natural language output. Algorithm A.7 classifies the energy intake of a proposed meal in one of the following five cases: *permanently inconsistent* (I.1), *occasionally inconsistent* (I.2), *consistent and not balanced* (C.1), *consistent and well-balanced* (C.2) and *consistent and perfectly balanced* (C.3).

In the cases C.1, C.2 and C.3 the value of the energy supply is consistent with the user's diet as represented in the STP. As in Algorithm A.2, it is possible to detect that a meal is consistent by exploiting the minimal network of the STP. A not balanced meal, even if consistent, will require to be compensated. We assume that the mean value is the "ideal" value according to the diet's goals and we use the difference $|\frac{a+b}{2} - EnergyIntake|$ as a simple distance for the macronutrient. To discriminate between the cases C.1, C.2 and C.3, we use two parametric user-adjustable values, t_1 and t_2, which set two thresholds relative to the mean value of the related STP constraints (see Fig. 4). In other words, we use the distance from the ideal value to classify the energy intake of the meal as not balanced (C.1), well balanced (C.2) or perfectly balanced (C.3) .

In the cases I.1 and I.2 the energy supply of the meal is inconsistent with

the diet. In the case I.1 the energy supply is inconsistent with regard to the user's diet as represented in the initial STP. This case is detected by considering whether the nutritional value of the meal violates an STP constraint. In the case I.2 the meal per se does not violate the STP constraint, but – considering the past meals the user has eaten – it would preclude to be consistent with the diet. Thus, it is inconsistent now, but in the future, e.g., next week, it could become possible to choose it. This case is detected by determining whether the energy supply, despite it satisfies the constraints in the initial STP, which does not consider the food intakes, is inconsistent with the STP constraint in the minimal network that takes into account also the previous food intakes.

Choice of the best meal. Such a facility, provided by Algorithm A.6 in Appendix A, can be employed when users must choose among a set of meals the best meal with regard to their diets.

Up to this point we defined the reasoning facilities considering only one STP at a time, and we had been able to disregard the fact that food actually has, at the same time, different quantities of the different macronutrients. Thus, when we deal, as in this reasoning facility, with the choice of a meal among a possibly limited set of available meals, we are not free to choose any hypothetically possible meal, but we have to choose the best one actually available by "integrating" the information provided by the three STPs corresponding to the macronutrients. To quantify the adequacy of an actual specific meal with respect to a diet, we need to define a multidimensional measure that balances the caloric distance for each STP. We define the Euclidean distance $d(ProposedMeal, MinNetMeal)$ between a $ProposedMeal$ and an "ideal" one represented by $MinNetMeal$, where the values of $MinNetMeal$ are obtained by identifying in the three minimal networks the constraints corresponding to the proposed meal. We compute such distance by considering the means of the

constraint values in the three minimal networks. More formally:

$$d(ProposedMeal, MinNetMeal) =$$
$$= \sqrt{\sum_{i \in \{Car,Pro,Lip\}} (ProposedMeal_i - \frac{MinNetMeal_{i,min} + MinNetMeal_{i,max}}{2})^2} \quad (1)$$

where $ProposedMeal_i$ is the caloric value for carbohydrates, proteins and lipids of a meal and $MinNetMeal_{i,min}$ and $MinNetMeal_{i,max}$ are the minimum and maximum caloric values of the corresponding constraints of the minimal networks of each macronutrient.

While in the food domain the STPs represent homogeneous values (i.e., kcal) and they can be combined as in formula (1), in general cake&carrot problems can have heterogeneous dimensions (for instance, steps taken and hours of sleep). In this more general case, to combine the different STPs each dimension should be adjusted by considering a proper weight.

Algorithm A.6 in Appendix A uses the distance in formula (1) to identify the meal, among the ones available, closest to the ideal value.

Note that such an algorithm makes choices that are locally optimal but are not guaranteed to be globally optimal, in the sense that at the end of the week the meals chosen by using Algorithm A.6 might not be the ones that minimize the distance with the diet. However, a globally optimal choice would require to know in advance the meals in the subsequent days. In Section 4 we experimentally evaluate the validity of the algorithm in a real-world scenario.

4. Results

In this section we describe a simulation designed to validate the approach, i.e., to verify the feasibility of the STP reasoning by comparing its performance with meaningful baselines in a realistic scenario. We use a computer simulation to primarily show that STP reasoning is robust with respect to dietary transgressions.

Here our main goal is to answer to the following research questions related to the STP reasoning:

- Can the STP Diet Reasoner effectively provide solutions for a healthy diet relying on a real database of dishes and real menus?
- Is the STP model adequate to reason about energy requirements and energy supply? What is the relation among diet constraints, users' energy requirements and meal composition in satisfying the user's diets?
- How much do the episodes of dietary transgression influence the achievement of the target behavior? How effective is our system in recovering from such episodes?

In order to answer to these questions we designed a simulation in a hospital context. We considered the situation where patients have to stay one week in the hospital and have to decide the dishes of each meal considering the hospital menu. Our system suggests the dishes composing a meal and the users can accept or reject such a suggestion thus composing a meal of their own choice. The hospital provides a real but controlled scenario for three reasons:

- The patients can only eat the hospital food, which is served for breakfast, lunch and supper.
- The patients can compose their meals choosing among a variable number of combinations of dishes. However, breakfasts, lunches and suppers follow three precise schemata, i.e., they are composed by a fixed number of dishes, belonging to specific categories (e.g., pasta, meat course).
- The energy supply of each dish is precisely known. Thus, in the simulation we do not account for indeterminacy in the calorie amount of a meal.

We have an ongoing collaboration with the Hospital "Città della Salute, Molinette" (Molinette henceforth) in Turin, which is the third largest hospital in Italy. For the simulation we used the menu that was served at Molinette in

June 2016. The hospital has a regular menu and 12 special menus addressing specific needs such as vegetarian, gluten-free, low-proteins, etc. In this experiment we considered the regular menu. The recipe database is composed by 246 different recipes along with their nutritional values. Since each dish is served in standard-size servings, we know in advance the nutritional value of each menu. As a working hypothesis, we assume that the patients eat their whole serving or do not eat the dish at all; in other words, we neglect the possible leftovers. However the impact of this simplification is limited by the fact that a meal is composed by different dishes and the user is free to reject any single dish.

In the Molinette regular menu, a meal has traditional Italian courses. Lunch and supper have a first course, usually pasta or soup (the hospital allows a patient to choose among seven or eight first courses, different each day of the week), a second course, usually meat, fish or cheese (a patient can choose among six dishes), a side dish (with four different dishes), fruit or dessert (five different dishes) and bread. A breakfast is composed by tea or coffee, milk, "fette biscottate" (a sort of rusk) or bread, butter, jam or honey. Thus, considering that a patient can also turn down a dish, there are 1680 different meals for lunch and 3780 different meals for dinner and 108 different meals for breakfast. Considering the different menus in the different days, a breakfast supplies between 377 kcal and 463 kcal, a lunch between 600 kcal and 1571 kcal, and a supper between 599 kcal and 1255 kcal.

In the simulation we considered six representative patients: three men and three women. Two men and two women are close to their normal weights, one man is obese and one woman is underweight. The six patients have different ages, heights and levels of activities. Upon a patient's weight, gender and age, using Schofield equation [11], it is possible to estimate a patient's basal metabolic rate as reported in Table 1. For the levels of activities we have taken into account that the patients are in a hospital and we have adjusted the standard levels of activities as in [12]; thus, the levels of activities range from 1.175 (i.e., bedrest) to 1.275 (i.e., mobilising occasionally on ward). Using Schofield equation we determined the users' daily energy requirements. Fur-

Age (years)	Basal metabolic rate (kcal/day)
Men	
$18-29$	$((0.063 \cdot weight) + 2.896) \cdot 238.8459$
$30-59$	$((0.048 \cdot weight) + 3.653) \cdot 238.8459$
> 60	$((0.049 \cdot weight) + 2.459) \cdot 238.8459$
Women	
$18-29$	$((0.062 \cdot weight) + 2.036) \cdot 238.8459$
$30-59$	$((0.034 \cdot weight) + 3.538) \cdot 238.8459$
> 60	$((0.038 \cdot weight) + 2.755) \cdot 238.8459$

Table 1: Schofield equation for estimating the basal metabolic rate [11].

User	Weight (kg)	Height (cm)	Age (years)	Gender	Level of activity	Body mass index (kg/m^2)	Transgression inclination	Energy requirement (kcal/day)
P1	73	175	40	M	1.175	23.8 (normal weight)	overnutrition	2009
P2	58	165	20	F	1.200	21.3 (normal weight)	alternating	1614
P3	70	175	30	M	1.275	22.9 (normal weight)	alternating	2136
P4	55	160	30	F	1.175	21.5 (normal weight)	undernutrition	1518
P5	92	175	30	M	1.200	30.1 (obese)	overnutrition	2313
P6	43	160	18	F	1.275	16.8 (underweight)	undernutrition	1432

Table 2: Patients in the simulation.

thermore, we reported the patients' inclination in dietary transgression, i.e., whether the patients, when they do not follow the system's suggestions, tend towards overnutrition, undernutrition or have no specific inclination. Patients' data are reported in Table 2.

At the beginning of the week, for each patient we computed the patient's energy requirement over a period of one week by using the Schofield equation [11] (see Section 3.1). As described in Section 3.1, we initially built three distinct STPs corresponding to the energy requirements and to the dietary prescriptions for the meals in the week for the three macronutrients: carbohydrates, proteins and lipids. In the initial values of the constraints, we admit three distinct tolerance values in the deviation from the ideal values, i.e., σ_w, σ_d and σ_m (see Section 3.1). In accordance with [9], we used a value of $\sigma_w = 10\%$ and $\sigma_d = 30\%$ for the tolerance values over a week and over a day since it they are reasonable

values that a dietitian can assume when assigning a diet to a patient. As a consequence, we used the value of $\sigma_m = 50\%$ for a meal. Thus, patients can make big deviations – however within the narrow limits imposed by the hospital menus – in the single meals, but they must follow their diets only with slight deviations over the week.

Once the patient actually eats her meal, the STPs are modified to take into account such a piece of information. Then, by means of constraint propagation, the system infers how the future meals are affected by such a fact.

To model the patient behavior and to validate the application of the STP approach, we introduce two policies: $STPDiet$ and $LocallyOptimalDiet$. In the policy $STPDiet$, based on the STP reasoner, the system suggests to the patient, among the compatible meals in the hospital menu, the one that "better satisfies" the STP constraints, i.e., the system suggests the meal with the minimum distance d from the mean between the minimum and maximum values of the meal's STP constraints (see the distance definition (1) in Section 3.4). Note that the compatibility of a meal means that the meal's macronutrients are consistent with the three STPs corresponding to the macronutrients. In the policy $LocallyOptimalDiet$, used as a baseline, the system suggests to the user the meal that "better satisfies" the diet requirements, i.e., the meal that has the closest energy supply to the ideal one that was decided for that meal at the start of the week. It should be noted that, if in the policy $STPDiet$ the STP constraints were not propagated, policy $LocallyOptimalDiet$ would be equivalent to policy $STPDiet$.

To evaluate how our system reacts to episodes of dietary transgression, we introduce a *transgression rate*. The transgression rate is the probability that, when users decide the composition of their meals, they do not follow the indications of the system and instead they eat other dishes in the hospital menu. In this case such a meal composition is chosen randomly (a) only among the compatible meals more caloric than the ideal one if the user has an inclination to *overnutrition*, (b) only among the compatible meals less caloric than the ideal one if the user has an inclination to *undernutrition* and (c) among all compatible

meals if the user has no specific inclination in her dietary transgressions. The transgression rate varies from 0%, i.e., the user always eats the indicated meal, to 50%, i.e., the user disobeys half the times. Transgressions can be performed independently of each other; in other words, the patients are free to transgress many times, even when their diets have been adjusted for compensating previous transgressions.

Since the transgression rate involves some random choices, we repeated the simulation 1000 times by changing the pseudorandom seed in each execution and we computed the averaged results and the 95% confidence interval.

In Table 3 we report the experimental results. In columns 1 and 2 we report the patient and the policy. In column 3 we report three levels of transgression rate (the results for transgression rates 10%, 30% and 40% are similar). At the end of each simulation, we evaluated the policies computing the distance (in kilocalories) between the user's weekly energy requirement, which is considered the ideal value provided by the diet, and the weekly calorie count. Formally, we define this distance as:

$$d\_eval = \left| \left(\sum_{M \in Meals} \sum_{i \in \{Car, Pro, Lip\}} energyIntake(M_i) \right) - DailyEnergyReq \cdot 7 \right| \tag{2}$$

In column 4 we report the average distance (2) from the ideal value and its 95% confidence interval. In column 5 we report, for the policy $STPDiet$, the difference with the results obtained by the policy $LocallyOptimalDiet$ considering the same transgression rate.

A number of interesting points arise from data:

- In all runs the $STPDiet$ policy reaches the end of the week without inconsistencies. This shows that our approach is actually applicable to realistic menus taken from real-world scenarios.

- The $STPDiet$ policy outperforms in a statistically significant way the $LocallyOptimalDiet$ policy for all patients and the difference between $STPDiet$ and $LocallyOptimalDiet$ policies grows up proportionally with

Patient	**Policy**	**Transgression rate**	**Distance from ideal value (95% confidence interval)**	**Δ**
P1	*LocallyOptimalDiet*	0%	139 (139-139)	
		20%	278 (270-285)	
		50%	444 (434-454)	
	STPDiet	0%	117 (117-117)	-22
		20%	253 (245-260)	-25
		50%	412 (403-421)	-32
P2	*LocallyOptimalDiet*	0%	188 (188-188)	
		20%	312 (303-321)	
		50%	503 (489-518)	
	STPDiet	0%	91 (91-91)	-97
		20%	223 (215-231)	-89
		50%	421 (408-434)	-82
P3	*LocallyOptimalDiet*	0%	313 (313-313)	
		20%	527 (517-537)	
		50%	817 (801-833)	
	STPDiet	0%	360 (360-360)	+47
		20%	527 (517-537)	0
		50%	782 (768-797)	-35
P4	*LocallyOptimalDiet*	0%	222 (222-222)	
		20%	412 (400-424)	
		50%	735 (714-756)	
	STPDiet	0%	169 (169-169)	-53
		20%	316 (305-326)	-96
		50%	478 (463-493)	-257
P5	*LocallyOptimalDiet*	0%	320 (320-320)	
		20%	471 (460-481)	
		50%	672 (657-687)	
	STPDiet	0%	227 (227-227)	-93
		20%	392 (381-403)	-78
		50%	627 (611-643)	-45
P6	*LocallyOptimalDiet*	0%	224 (224-224)	
		20%	485 (471-500)	
		50%	932 (910-954)	
	STPDiet	0%	175 (175-175)	-49
		20%	401 (389-414)	-84
		50%	597 (581-613)	-335

Table 3: Numerical results of the simulation with *LocallyOptimalDiet* and *STPDiet* policies.

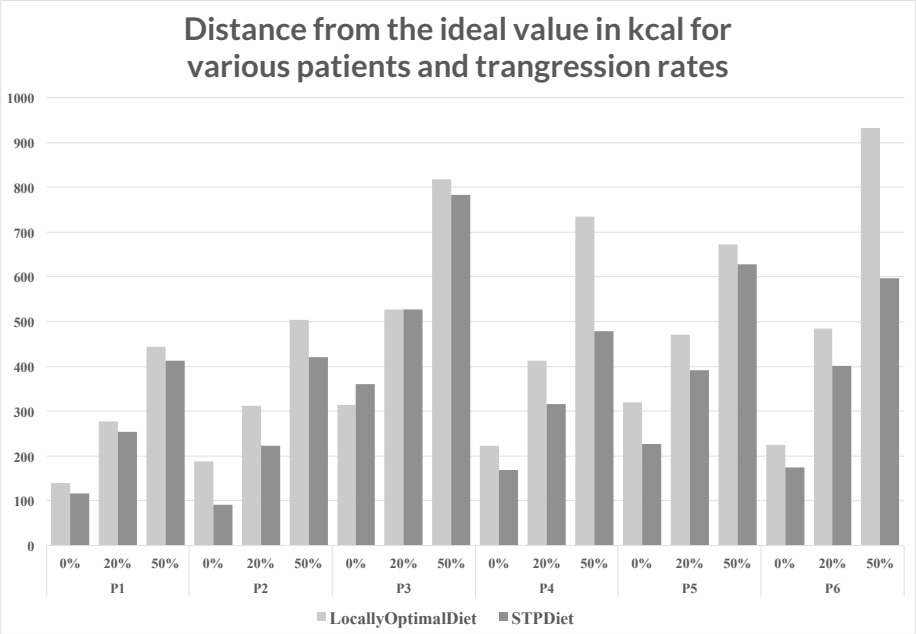

Figure 5: Comparing the results of the simulation with *LocallyOptimalDiet* and *STPDiet* policies (smaller is better).

the transgression rate for all the patients. The only exception is patient $P3$ with transgression rate 0.

In Fig. 5 we graphically compare the distances from the ideal values of the *LocallyOptimalDiet* and *STPDiet* policies. We can summarize the results depicted by noting that STP reasoning is always better than the other policy in the management of a diet with transgressions.

It is worth noting that, during the simulation, the choice of the meal is optimal with respect the measure (1), which, however, is not granted to reach the optimal final solution defined with respect measure (2). Indeed, a globally optimal decision should be informed about the future, i.e., it should take into account also the future meals. Moreover, note that we have used three distinct and independent STPs to represent the three macronutrients, but, when composing a meal, it is not possible to choose independently the three macronutrients. De-

spite these two critical issues, the reasoner evaluation in terms of the distance between the actual weekly calories consumption and its theoretical optimal value confirms that the STP-based reasoner overcomes the baseline guiding the user towards choices close to the optimal ones.

At this point we can answer to the research questions posed at the beginning of this section. We showed that the STP Reasoner provides good solutions for managing a healthy diet in a real context since the STP adequately models the energy requirements and the food eaten. Moreover, we showed that the STP model improves significantly the achievement of the diet goals in the case of dietary transgressions.

Limitations of the simulation experiment

In the simulation we relied on a number of working hypotheses that imply some limitations; the main limitations are related to the use of a hospital setting: in particular, (1) the adoption of a controlled menu, (2) the use of prototypical patients, and (3) the neglecting of the leftovers.

With respect to the adoption of a controlled menu, on the one hand it should be noted that the convenience for adjusting patients' menus according to their diets could emerge also in the hospital setting. Indeed, hospital menus are usually not customized for the specific needs of the single patients (with the exception of the special menus required by some specific examinations). On the other hand, in our simulation we had to stick to the hospital menu also for compensating possible imbalances, whereas, outside of the hospital, a much wider variety of food is available (e.g., users could choose a protein bar to boost their protein assumption). In this sense the hospital context is not a completely favorable environment for the simulation and we expect that in a everyday setting our system would be most useful.

With respect to the use of six prototypical patients, we note that these patients are not real human beings participating to a specific clinical experiment, but we have modeled their profiles by adopting distinct but realistic values for their physical properties. As a future work we plan to perform a real-world

validation with human patients.

Finally, we neglected the impact of the possible leftovers. We think that this simplification does not invalidate the results of the simulation; actually, our claim is that, if the leftovers are taken into account, the difference between the *STPDiet* policy and the *LocallyOptimalDiet* policy should become larger. In fact, on the one hand, the *LocallyOptimalDiet* policy does not dynamically adapt the patient's diet decided at the start of the week and, on the other hand, the *STPDiet* policy exploits the constraint propagation mechanism described so far. As a future work it could be interesting to integrate our system with methods that estimate the leftover amount such as [13].

5. Discussion and conclusions

In this paper we presented a reasoning system based on STP. In order to account for the dynamic nature of the diet-user interaction, we proposed a new theoretical framework which exploits and extends the STP framework. In particular, in Section 3 and in Appendix A we have defined nine different algorithms: while Algorithm A.1 is a direct consequence of the mathematical properties of STP, the remaining algorithms have been designed to extend the STP representation power. In fact, we considered, in addition to standard STP constraints, also indeterminate constraints. Moreover, we formalized a number of facilities exploiting the minimal network computed by the framework. In, particular, such facilities allow to: (1) check the consistency of a diet, (2) check the compatibility of a single meal with respect to a diet, (3) check the compatibility of several meals with respect to a diet, (4) selectively list the dietary constraints of a diet, (5) reason on the consequences of the choice of a specific food for the rest of the diet, (6) choose the best meal among a set of possibilities, (7) provide a symbolic evaluation of a proposed meal. In Section 4 we have presented a simulation based on realistic user profiles and real hospital data. The evaluation of the reasoner is based on the distance between the actual weekly calories consumption and its theoretical optimal value. The results of

the simulation confirm that our STP-based reasoner, which supports compensation of the transgressions, overcomes the baseline, which does not account for transgressions, and gives results close to the optimal ones.

Some early work dealt with the task of planning a diet by modeling the task as a linear programming problem and by solving it with Operational Research techniques as the simplex method (see the survey in [14] or, more recently, [15]). However, these approaches do not support the users in choosing a meal and in investigating the consequences of their choices, but they only plan an entire diet. In [10] Buisson devised a system employing fuzzy arithmetic and heuristic search for assessing the compatibility of a single meal to a norm and for suggesting to the users some actions such as removing or adding food to balance their meals. [10] supports fuzzy arithmetic to represent imprecision/uncertainty in quantity and composition of food, however it does not tackle the problem of globally balancing the meals. Other related works include both academic studies and commercial projects. Among academic studies related to our project we cite [16, 17, 18, 19], and among the smartphone apps related to nutrition we cite *DailyBurn, Lose It!, MyNetDiary, A low GI Diet, WeightWatchers*. With regard to all these works, our reasoning framework, when applied to the dietary domain, presents a big element of novelty: the use of automatic reasoning as a tool for verifying the compatibility of a specific meal with a specific diet, for suggesting a compatible meal, for determining the consequences of the choice of a specific dish and for recovering from episodes of dietary transgression.

Some ideas contained in this paper have been previously presented in [20], where we depicted a general architecture for the diet management. This paper is longer and significantly different; in particular, in contrast to [20], in this paper we propose the general class of cake&carrot problems, we model diet and food in terms of the three macronutrients whereas [20] uses only the total amount of energy intake, we propose a number of algorithms to enhance STP with a dynamic perspective, we discuss the correctness of the reasoning facilities, we support indeterminacy/imprecision in food intake, we provide an experimental quantitative evaluation by using a simulation based on real data. Moreover, in

contrast to [20], here we do not present the general architecture for diet management but focus on the reasoning system and we do not discuss the natural language generation module.

The reasoning framework presented in this paper can be commercially attractive for different applications in the quantified self domain. For instance, given the large diffusion of wearable tracking gadgets, the framework seems appealing for step counting and sleep time management. In the specific case of the diet domain we believe that there are at least in two contexts for application. The first context is the medical one, where users (e.g., patients affected by essential obesity) are strongly motivated to strictly follow a diet and need tools that help them. The second context is the one involving, e.g., healthy fast food or restaurant chains, where the effort of deploying the system can be rewarded by an increase in customer retention.

In the next future, we plan to experiment the system with a focus group in a clinical context with patients affected by essential obesity. In this way we can give a definitive validation of the reasoning framework that at this time has been tested only in a simulated environment. In this clinical setting we expect that our system can also support human dietitians in the supervision of their patients.

Acknowledgment

This research has been partially funded by Regione Piemonte, Innovation Hub for ICT, 2011-2014, POR-FESR 07-13. The authors would like to thank the editors and the anonymous reviewers for their valuable comments and suggestions that allowed to improve the quality of the paper.

References

[1] T. Abdel-Hamid, F. Ankel, M. Battle-Fisher, B. Gibson, G. Gonzalez-Parra, M. Jalali, K. Kaipainen, N. Kalupahana, O. Karanfil, A. Marathe, B. Martinson, K. McKelvey, S. N. Sarbadhikari, S. Pintauro, P. Poucheret,

N. Pronk, Y. Qian, E. Sazonov, K. Van Oorschot, A. Venkitasubramanian, P. Murphy, Public and health professionals' misconceptions about the dynamics of body weight gain/loss, System Dynamics Review 30 (1-2) (2014) 58–74.

[2] R. Dechter, I. Meiri, J. Pearl, Temporal constraint networks, Artif. Intell. 49 (1-3) (1991) 61–95.

[3] T. Vidal, Handling contingency in temporal constraint networks: from consistency to controllabilities, Journal of Experimental & Theoretical Artificial Intelligence 11 (1) (1999) 23–45.

[4] P. H. Morris, N. Muscettola, Temporal dynamic controllability revisited, in: AAAI, 2005, pp. 1193–1198.

[5] A. Cimatti, L. Hunsberger, A. Micheli, R. Posenato, M. Roveri, Sound and complete algorithms for checking the dynamic controllability of temporal networks with uncertainty, disjunction and observation, in: 2014 21st International Symposium on Temporal Representation and Reasoning, IEEE, 2014, pp. 27–36.

[6] C. Nishida, R. Uauy, S. Kumanyika, P. Shetty, The joint WHO/FAO expert consultation on diet, nutrition and the prevention of chronic diseases: process, product and policy implications, Public Health Nutrition 7 (2004) 245–250. doi:10.1079/PHN2003592.

[7] LARN - Livelli di Assunzione di Riferimento di Nutrienti ed energia per la popolazione italiana - IV Revisione, SICS Editore, Italy, 2014.

[8] L. Fontana, S. Klein, Aging, adiposity, and calorie restriction, JAMA 297 (9) (2007) 986–994. doi:10.1001/jama.297.9.986.

[9] S. A. Bingham, The dietary assessment of individuals: Methods, accuracy, new techniques and recommendations, Nutr Abstracts Rev 57 (1987) 705–742.

[10] J.-C. Buisson, Nutri-educ, a nutrition software application for balancing meals, using fuzzy arithmetic and heuristic search algorithms, Artif. Intell. Med. 42 (3) (2008) 213–227.

[11] W. N. Schofield, Predicting basal metabolic rate, new standards and review of previous work, Human Nutrition: Clinical Nutrition 39C (1985) 5–41.

[12] S. Ferrie, M. Ward, Back to basics: Estimating energy requirements for adult hospital patients, Nutrition & Dietetics 64 (3) (2007) 192–199.

[13] G. Ciocca, P. Napoletano, R. Schettini, New Trends in Image Analysis and Processing – ICIAP 2015 Workshops: ICIAP 2015 International Workshops, BioFor, CTMR, RHEUMA, ISCA, MADiMa, SBMI, and QoEM, Genoa, Italy, September 7-8, 2015, Proceedings, Springer International Publishing, Cham, 2015, Ch. Food Recognition and Leftover Estimation for Daily Diet Monitoring, pp. 334–341.

[14] L. M. Lancaster, The history of the application of mathematical programming to menu planning, European Journal of Operational Research 57 (3) (1992) 339 – 347.

[15] E. Bas, A robust optimization approach to diet problem with overall glycemic load as objective function, Applied Mathematical Modelling 38 (19–20) (2014) 4926 – 4940.

[16] J. L. Balintfy, Menu planning by computer, Commun. ACM 7 (4) (1964) 255–259.

[17] K. Iizuka, T. Okawada, K. Matsuyama, S. Kurihashi, Y. Iizuka, Food menu selection support system: considering constraint conditions for safe dietary life, in: Proceedings of the ACM multimedia 2012 workshop on Multimedia for cooking and eating activities, CEA '12, ACM, New York, NY, USA, 2012, pp. 53–58.

[18] J. Mankoff, G. Hsieh, H. C. Hung, S. Lee, E. Nitao, Using low-cost sensing to support nutritional awareness, in: Proceedings of the 4th international

conference on Ubiquitous Computing, UbiComp '02, Springer-Verlag, London, UK, UK, 2002, pp. 371–376.

[19] K. A. Siek, K. H. Connelly, Y. Rogers, P. Rohwer, D. Lambert, J. L. Welch, When Do We Eat? An Evaluation of Food Items Input into an Electronic Food Monitoring Application, in: Pervasive Health Conference and Workshops, 2006, 2006, pp. 1–10. doi:10.1109/pcthealth.2006.361684.

[20] L. Anselma, A. Mazzei, Towards Diet Management with Automatic Reasoning and Persuasive Natural Language Generation, in: Progress in Artificial Intelligence - 17th Portuguese Conference on Artificial Intelligence, EPIA 2015, Coimbra, Portugal, September 8-11, 2015. Proceedings, 2015, pp. 79–90. doi:10.1007/978-3-319-23485-4_8.

Appendix A.

Algorithm A.1 Computing the minimal network of a diet

function MinimalNetworkOfADiet($DietSTP$)
    $DietMN \leftarrow FloydWarshall(DietSTP)$
    if $DietMN$ has a negative cycle **then**
        return $Inconsistent$
    else
        return $DietMN$

Algorithm A.2 Compatibility check of a single meal with respect to a diet

function CompatibilityCheckOfASingleMeal($MinimalNetwork$, $\langle Day, MealType, EnergyIntake \rangle$)
    let $a \leq E - S \leq b$ the STP constraint in $MinimalNetwork$ over the starting and ending points of $MealType$ in Day
    if $a \leq EnergyIntake \leq b$ **then**
        return $True$
    else
        return $False$

Algorithm A.3 Computing the minimal network of a diet with respect to the meals taken

function MinimalNetworkOfADietWithFood($DietSTP$, $\langle Day_1, MealType_1, EnergyIntake_1\rangle$, ..., $\langle Day_n, MealType_n, EnergyIntake_n\rangle$)
  let $integratedDietSTP$ be a provisional copy of $DietSTP$
  for each $(MealType_i, Day_i)$ **do**
    let $a \leq E - S \leq b$ the STP constraint in $DietSTP$ over the starting and ending points of $MealType_i$ in Day_i
    substitute in $integratedDietSTP$ the STP constraint over $MealType_i$ in Day_i with the constraint $max(a, EnergyIntake_i) \leq E - S \leq min(b, EnergyIntake_i)$
  $integratedDietMN \leftarrow$ FloydWarshall($integratedDietSTP$)
  if $integratedDietMN$ has a negative cycle **then**
    return $Inconsistent$
  else
    return $integratedDietMN$

Algorithm A.4 Simulation of several meals with respect to a diet

function WhatIf($MinimalNetwork$, $\langle Day_1, MealType_1, EnergyIntake_1\rangle$, ..., $\langle Day_n, MealType_n, EnergyIntake_n\rangle$)
  let $MinimalNetwork'$ be a provisional copy of $MinimalNetwork$
  for each $(MealType_i, Day_i)$ **do**
    let S and E be the starting and ending points of $MealType_i$ in Day_i
    substitute in $MinimalNetwork'$ the STP constraint between S and E with the constraint $EnergyIntake_i \leq E - S \leq EnergyIntake_i$
  $MinimalNetwork' \leftarrow$ FloydWarshall($MinimalNetwork'$)
  List the dietary constraints in $MinimalNetwork'$

Algorithm A.5 List the dietary constraints represented in the minimal network

function ListDietaryConstraints($MinimalNetwork$)
  for each day Day_i **do**
    for each meal $MealType_j$ **do**
      let $a \leq E - S \leq b$ the STP constraint in $MinimalNetwork$ over the starting and ending points of $MealType_j$ in Day_i
      represent $a \leq E - S \leq b$
    let $a \leq E - S \leq b$ the STP constraint in $MinimalNetwork$ over the starting and ending points of Day_i
    represent $a \leq E - S \leq b$
  let $a \leq E - S \leq b$ the STP constraint in $MinimalNetwork$ over the starting point of the first day and the ending point of the last day
  represent $a \leq E - S \leq b$

Algorithm A.6 Choose the best meal with respect to a diet

1: **function** ChooseBestMeal($MinimalNetwork$, $\langle Day, MealType \rangle$, $\{EnergyIntake_1, \ldots, EnergyIntake_n\}$)
2: let $a \leq E - S \leq b$ the STP constraint in $MinimalNetwork$ over the starting and ending points of $MealType$ in Day
3: **for each** $EnergyIntake_i$ **do**
4: Compute the distance between $a \leq E - S \leq b$ and $EnergyIntake_i$
return the closest $EnergyIntake_i$

Algorithm A.7 Evaluate a meal with respect to a diet

1: **function** EvaluateMeal($DietSTP$, $MinimalNetwork$, $\langle Day, MealType, EnergyIntake \rangle$)
2: let $a \leq E - S \leq b$ the STP constraint in $MinimalNetwork$ over the starting and ending points of $MealType$ in Day
3: Compute the distance between $a \leq E - S \leq b$ and $EnergyIntake$
4: **if** distance is between 0 and t_1 **then return** "C3. Perfectly balanced"
5: **else if** distance is between t_1 and t_2 **then return** "C2. Well balanced"
6: **else if** distance is greater than t_2 and $EnergyIntake$ is between a and b **then return** "C1. Not balanced"
7: **else**
8: let $a' \leq E - S \leq b'$ the STP constraint in $DietSTP$ over the starting and ending points of $MealType$ in Day
9: **if** $EnergyIntake$ is not between a and b but it is between a' and b' **then return** "I2. Occasionally inconsistent"
10: **else**
11: **return** "I1. Permanently inconsistent"