smartfit: Using Knowledge-based Configuration for Automatic Training Plan Generation

Florian Grigoleit, Peter Struss, Florian Kreuzpointner
Technische Universität München
Boltzmannstr. 3, 85748 Garching b. München
{grigolei, struss}@in.tum.de, kf@tum.de

Abstract

The fitness industry has been booming for several decades, and there is an increasing awareness of the essential impact of physical exercise on health. Those who are interested in exercising usually lack detailed knowledge about how to do this in a way that is effective and appropriate. Existing apps mainly offer a set of standard training plans that do not take all relevant individual and contextual conditions into account. The resulting effect of following these apps may not only be ineffective, but even harmful to health. Properly designed training plans, as usually produced by an experienced trainer, must consider both individual goals and physical abilities of the trainees to avoid adverse effects. We developed s*martfit* as a knowledge-based system for generating training plans tailored to the individual trainee without requiring detailed knowledge. It has been developed as an application of our generic constraint-based configuration system GECKO, which generates optimal or optimized configurations that satisfy high-level user demands. We briefly introduce GECKO, present the application problem and the domain knowledgebase, and discuss the evaluation of the current system and future work.

1 Introduction

Creating a training plan at home appears to be simple. A trainee chooses exercises and performs them. Usually, such an approach results in unsatisfactory training results. First, an average trainee lacks the necessary training knowledge. Second, background knowledge regarding health and training effects, implicitly included in a professional training plan, is either unavailable to the average trainee or too complex for him/her to include it in a training plan. For these reasons, homemade training plans tend to be insufficient. The same applies to most training plans available on the internet, which just consider very few parameters like gender and training goal. This leads to an unsatisfactory training plan, which does not reflect the needs of the trainees.

To provide trainees with effective, customized training plans with a positive impact on health, we developed a knowledge-based solution based on GECKO (Generic, constraint-based Konfigurator), [9]: *smartfit*. s*martfit* is designed for trainees who want to create plans based on deep background knowledge and which cover the needs for individual personalized expectancies.

Creating a good training plan is a very complex task, consisting of selecting and parameterizing exercises based on user parameters and domain knowledge. This is analogous to configuring a system based on a repository of components (which are usually physical building blocks or software modules), [16], and, therefore, we base *smartfit* on GECKO. Together with researchers from sports and health sciences, we created a descriptive domain theory for fitness training. This domain theory is a specialization of generic GECKO concepts and a collection of constraints on their attributes.

In this paper, we focus on presenting the solution to configuring a plan for a single training session based on an initial version of the knowledge base. Section 2 introduces training science and motivates our work on generating training plans automatically. Next, we introduce our formalization of the configuration task and the key concepts of GECKO. The knowledge representation of fitness training is described in Section 4, while Section 5 evaluates the solution. Finally, we comment on our current work and some open issues.

2 Generation of Training Plans

Training science is a discipline of sport sciences focused on analyzing the effects of training stimuli on the human body. The effects of training can vary significantly. It can enhance aerobic capacity, increase flexibility, or improve strength abilities. Trainees can have several reasons for training, but all have one goal in common: they want to enhance their physical performance. One major insight of training science is that adaptation to training is highly individual regarding the trainee. The same training stimulus can have different effects on different trainees depending on the individual physiological capacity. Therefore, it is very important to train under optimal conditions with an appropriate training plan to have individual success. For this, a trainer has to select a set of exercises, the load (training weight) of the exercises, and the amount of rest between the sets and exercises. To create a plan for an individual trainee, he must consider parameters such as age or the individual fitness of the trainee, because an intensive exercise e.g. *burpees*, is well-suited for young and fit trainees, but would overwhelm and potentially even harm beginners or elderly trainees. With this information, two important pillars are covered: What should I train, and how?

Elite athletes perform highly individualized training, which makes them stronger and better. This is only possible because their trainers, scientific and medical advisors etc. possess the required specific knowledge. Common trainees do not have access to this knowledge. Therefore, with our work, we aim at collecting it in a knowledge-base and making it exploitable in the generation of individual training plans without requiring the user to acquire the detailed domain knowledge him/herself. With this, we expect to contribute to making exercising more effective and satisfactory to sports amateurs and occasional practitioners. In this way, the trainees may avoid frustration and adverse results.

The starting point for creating a training plan is considering the aim of the trainee and break it down into desired improvements in various fitness categories, for example strength or endurance. In addition, further information about the trainee and the training conditions is necessary: age, height, weight, and gender as well as information about available equipment and training duration are required. Further questions include: are there any health or injury concerns? What is the desired or available training frequency per week? Which muscle groups have to or are desired to be or trained?

Training planning includes arranging the exercises within a session appropriately (obviously, the warm-up should be before the main training). However, despite the term "planning", no elaborate planning in the sense of producing complex structural and temporal interdependencies of actions is required. The main task is selecting exercises from a repository and parameterizing them. Ordering them in an appropriate way is usually not a major problem and will not lead to the rejection of a set of exercises.

3 A Generic Knowledge-based Approach to Generating Optimized Configurations

The main objectives of our development of GECKO are

- a fairly **domain-independent** solution to configuration problems,
- based on a small set of **generic concepts** that support a clear structuring of the knowledge base,
- allowing its use **without** detailed **domain knowledge,**
- considering **optimality** criteria.

The creation of a knowledge base for a specific application system is done by providing

- domain specific **specializations** of the generic concepts and specification of variables associated with them and
- **constraints** of different types on these variables.

We first briefly introduce the concepts structuring the knowledge-base, using smartfit to illustrate them, and then present the theoretical and algorithmic foundations (for more information, see [9]).

3.1 GECKO Concepts

The three key concepts underlying the system have a straightforward intuitive meaning.

- **Component**: the elements to be chosen and included in a configuration (in smartfit: exercises),
- **Goal**: achievements expected from a configuration; they can express high-level user expectations ("muscle gain") or detailed sub-goals the user is unaware of ("biceps hypertrophy")
- **Task**: requirements and restrictions on the resulting configuration (including at least one goal specified by the user), such as available training equipment and physical properties of the trainee.

Domain knowledge is expressed by constraints on (attributes of) subclasses or instances of these concepts. Figure 1 displays the different types of constraints. For instance, certain parameters characterizing the task may by incompatible with certain goals ("low body-mass-index excludes goal weight-loss") or exclude some components (an injury may prevent certain exercises). There are interdependencies among components (two particular exercises must not appear together in a session) or goals, which can be used to introduce a decomposition of goals ("muscle gain" requires "muscle gain of upper body" and "muscle gain of lower body"). GECKO offers the basic constraints "requires" (implication) and "excludes" (implication of negation) to represent this.

A key part of the configuration knowledge is related to the proper selection of components given the goals stated in or resulting from the task. GECKO uses the construct of **Choice** to represent this: it is a collection of components each with an associated **Contribution**, a numerical or qualitative value. The contributions of the components included in a configuration are summarized by the choice to deliver a certain reached **AchievementLevel**. The goal which has this choice associated has an **AchievementThreshold** that needs to be reached by the **AchievementLevel** in order to be considered fulfilled.

Since a component may occur in several choices (an exercise affects several muscle groups), we obtain an m:n relationship between goals and components (and introduce the potential of a combinatorial problem). GECKO also uses choices of goals to express how sub-goals together achieve a higher-level goal.

Finally, goals may have an associated priority, which assures that the more important goals receive more contributions, and components have a cost. One important task parameter is a limit to the cost of the entire configuration, which typically (but not necessarily) is the sum of the components' costs. While cost will often really mean "money", in the training plan domain, time is the resource which is limited and consumed by the exercises.

Contributions and cost are the factors that allow characterizing the utility of selecting a component and, hence, for specifying optimal solutions and for guiding a best-first search.

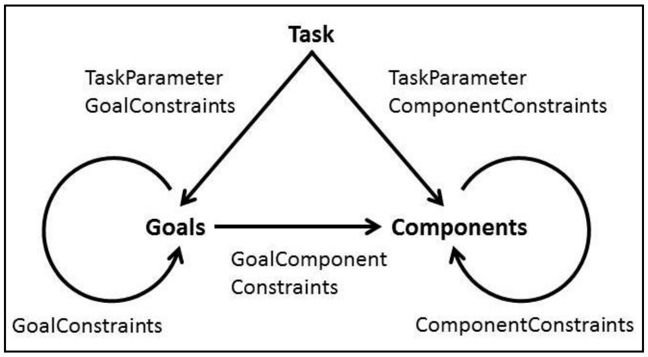
Task
TaskParameter GoalConstraints
TaskParameter ComponentConstraints
Goals
Components
GoalComponent Constraints
GoalConstraints
ComponentConstraints

Figure 1 Constraint for configuration

3.2 Consistency-based Configuration

We formalize the configuration problem as identifying a subset of the components that satisfies the task specified by the user and is consistent with the configuration knowledge base **ConfigKB.** This can be seen as an assignment, AA, of activity to the components, which indicates the inclusion in or exclusion from the configuration.

Definition 1 (Activity Assignment)
An activity assignment for a set $COMPS_0 \subseteq COMPS$ is the conjunction $AA(COMPS_0)$ =

$$\left[\bigwedge_{comp \in COMPS_0} ACT(comp)\right] \wedge \left[\bigwedge_{comp \in COMPS \setminus COMPS_0} \neg ACT(comp)\right]$$

ACT(comp) is a literal which holds when a component $comp \in COMPS$ is part of a configuration.

Definition 2 (Configuration Task)
A configuration task is a pair (ConfigKB, Task) where:

- *ConfigKB, the knowledge base, containing the domain-specific objects and constraints,*
- *Task is a triple (TaskGoals, TaskParameters, TaskRestrictions) where:*
    - *TaskGoals, is the assignment of goal.Achieved=T to a set of user selected goals a solution to a configuration problem has to satisfy*
    - *TaskParameters, domain-specific value assignments to parameters, are constants*
    - *TaskRestriction, user selected constraints on the activity of components*

To establish a solution to a configuration task, a set of active components has to be consistent with the task and the knowledge base.

Definition 3 (Configuration for a Task)
A configuration for a Task is an activity assignment $AA(\Gamma)$ such that $ConfigKB \cup TASK \cup \{ AA(\Gamma) \}$ is satisfiable.

$$ConfigKB \cup TASK \cup \{ AA(\Gamma) \} \nvDash \perp$$

A configuration is minimal iff for no proper subset Γ' of Γ is $AA(\Gamma')$ is a configuration.

Consistency seems to be a weak condition. After all, we want the configuration to satisfy the goals, not just be consistent with them. But this is ensured by the definition, as stated by the following proposition. Intuitively, if an activity assignment yields an AchievementLevel lower than the AchievementThreshold of a goal, it would be inconsistent with the goal.Achieved=T as required by the task.

Proposition 1
If $AA(\Gamma)$ is a solution to a configuration task (ConfigKB, Task) then

$$AA(\Gamma) \cup ConfigKB \vDash \forall\, goal \in TaskGoals\ goal.Achieved = T$$

This view on configuration was inspired by the formalization of consistency-based diagnosis [1], [7], where modes OK or ¬OK are assigned to the components of a system, and a diagnosis is defined as a mode assignment MA(Δ) that is consistent with the model library, the structural description of the system and a set of observations (which are all sets of constraints, just like ConfigKB and Task):

$$ModelLib \cup Structure \cup Obs \cup \{ MA(\Delta) \} \nvDash \perp$$

In consequence, solutions to consistency-based diagnosis can also be exploited for generating configurations. This includes the introduction of a utility function and the application of best-first search to generate solution.

3.3 Search for Optimal Configurations

In consistency-based diagnosis, a utility function is often based on probabilities of component modes (assuming independent failures of components) ([7]) or, weaker, some order on the modes ([13]). In GECKO, we consider the contributions of components to the satisfaction of goals (possibly weighted by priorities of goals) and their cost.

Definition 4 (Utility Function)
A function $h(AA(\Gamma)$, Task) is a utility function for a configuration problem iff it is admissible for A search.*

Definition 5 (Optimal configuration)
A configuration $AA(\Gamma)$ is optimal regarding a utility function $h(AA(\Gamma)$, Task) iff for no configuration $AA(\Gamma')$ $h(AA(\Gamma')$, Task)is larger.

The utility of a configuration represents the fulfillment of the required goals and the cost of the configuration.
The utility depends on its active components only. In the following, it is assumed that

- the contribution of a configuration is obtained solely as a combination of contributions of the active components included in the configuration and otherwise independent of the type of properties of the components,
- the cost of the contribution is given as the sum of the cost of the involved active components and will usually be numerical, and
- we can define a ratio "/" of contributions and cost.

The first defined function sums up the AchievementLevels (i.e. the combined contributions of all active components) multiplied with a weight dependent on the goal priority of all active goals and divides this by cost of all active components. (In the definition, we simplify the notation by writing $Goal_j$.AchievementLevel instead of $Goal_j.Choice_j$. AchievementLevel etc.).

Definition 6 (GECKO Utility Function)

$$hl(AA(\Gamma, ActGoals) := \frac{\sum_{Goal_j \in ActGoals} weight\left(Goal_j.Priority\right) * Goal_j.AchievementLevel}{\sum_{Comp_i \in \Gamma} Comp_i.Cost}$$

This function ignores an important aspect: If the AchievementThreshold of some choice has already been reached, the utility of adding yet another component with a contribution to this choice is overestimated. The second utility function tries to capture this by disregarding any excesses above the AchievementThresholds.

Definition 7 (GECKO Utility Function with contribution limit)

$$hl(AA(\Gamma, ActGoals) \coloneqq \frac{\sum_{Goal_j \in ActGoals} weight\left(Goal_j.Priority\right) * CurbedLevel(Goal_j)}{\sum_{Comp_i \in \Gamma} Comp_i.Cost}$$

Where CurbedLevel is defined by

$$CurbedLevel \coloneqq \min(Goal_j.AchievementLevel, Goal_j.AchievementThreshold)$$

Based on this, we can exploit best-first search and solutions that have been developed in the context of consistency-based diagnosis. This includes pruning the search space based on inconsistent partial mode assignments that have been previously detected during the search (called conflicts), e.g. exploiting a truth-maintenance system (TMS, such as the assumption-based TMS [6]) as in SHERLOCK does ([8]). Classical A* search has been extended and improved by. From the diagnostic solutions, this approach has been generalized later as conflict-directed A* search, see [9].

4 smartfit

In this section, we discuss the configKB for the domain of training plan generation. Of course, we can present only the basic principles and some typical examples for illustration purposes. We will provide some details on the scope and size of the knowledge base. The conceptualization and structure of the domain knowledge is nothing that can be extracted from a textbook or obtained directly from interviewing experts. It is the result of major knowledge acquisition efforts requiring several person years and involved sports scientists, professionals from the fitness business, and AI researchers. With this application, we support our claim that GECKO provides a basis for creating specific application systems by specializing the generic classes and providing a structured set of constraints, see Table 1. The presentation is restricted:

- to generating a plan for one training session
- as a set of exercises, i.e. without ordering them and
- without their parameterization.

4.1 smartfit's Essential Concepts

Goals

Goals in *smartfit* represent certain aspects or requirements a Trainee must fulfill to improve his/her fitness. The smartfit domain theory contains three hierarchically ordered types of goals:

- **TraineeGoals** are high-level goals selected by the user. They are also the only goals that the user has to be aware of. They represent an abstract achievement the user wants to achieve, e.g. *weight loss* or *muscle gain.*
- **TrainingGoals** represent a specific aspect of fitness training, e.g. strength training, under consideration of a TraineeGoal, i.e. strength training to support weight loss. The corresponding TrainingGoal is *WeightLoss.Strength.*
- **TargetGoals**: A TargetGoal represents a single fitness target, i.e. a body region or a muscle, to be trained. TargetGoals are RegionsGoals, MuscleGroupGoals, and MuscleGoals, see Figure 2 The intensity with which it is to be trained depends on the corresponding TrainingGoal.

Goals are organized in a hierarchical structure via requires constraints (Goal-Goal Constraints). Requires (x, y) is defined by

x.active=T => y.active=T,

for configuration constraints, see [9]. For example, the TraineeGoal MuscleGain would require the TrainingGoals MuscleGain.Strength and MuscleGain.Endurance. The TrainingGoals in turn require subordinate TargetGoals, as shown in Figure 4.

Table 1: Overview of specialized GECKO concepts

GECKO Concept	**Fitness Concept**	**Example**
Goal	TraineeGoal	Muscle Gain
	TrainingGoal	Strength
	TargetGoal	Biceps
Component	Exercise	Push-up
Task	Training Request	-
Task – Restriction	TrainingDuration	90 minutes
	ExerciseRestriction	Exclude(push-up)
Task – Parameter	TrainingProperty	Equipment
	TraineeProperty	FitnessTraget.Biceps
Configuration	TrainingPlan	

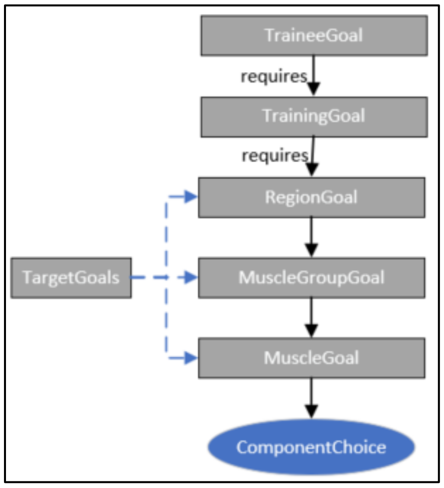

Figure 2: Goal-Structure for smartfit

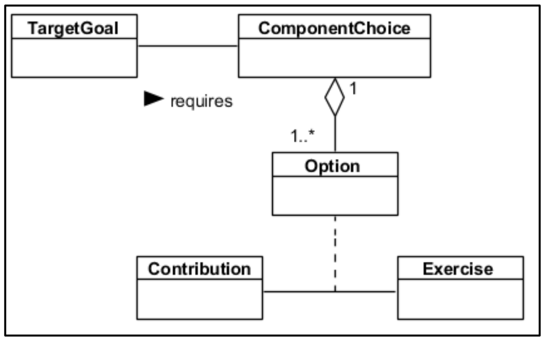

Figure 4: Expanding Goal-Structure for MuscleGain

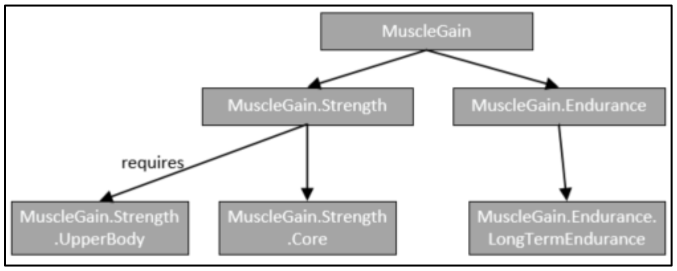
MuscleGain
MuscleGain.Strength
MuscleGain.Endurance
requires
MuscleGain.Strength .UpperBody
MuscleGain.Strength .Core
MuscleGain.Endurance. LongTermEndurance

Figure 3 Relation Goal-Choice-Component

The expanded Goal-Structure in Figure 4 shows the body regions UpperBody and Core as lowest level. The granularity of this structure depends on the associated TrainingGoal and, in some cases, on the TraineeGoal. While it is necessary for strength training to break down the body region into muscle groups, such as upper leg or abdominal region, and specific muscles, e.g. biceps or triceps, this is not the case for endurance training, for which either the entire body or body regions, e.g. legs, are sufficiently precise. What is important to note is that only the last level or TargetGoals is connected to exercises (components) via requiresChoice constraints.

As explained in [9], Goals can have priorities. In smartfit, priorities (domain={1,2,3,4,5}) indicate not (only) the importance of a goal, but the focus of the training. If the priority of the TrainingGoal *strength* is higher than the priority of *endurance*, more exercises and more time are required to achieve strength than for endurance.

The existing knowledge base contains 8 TraineeGoals, 24 TrainingGoals (3 fitness categories * 8 TraineeGoals), 72 RegionGoals (3 body regions * 24 TrainingGoals).
The priority of each goal is defined in the knowledge base. If the priority is changed, e.g. by the training focus (see Task), the increase/decrease is propagated downwards through the goal structure.

Components

Components in smartfit are Exercises. An Exercise is an activity in fitness training designed to train a FitnessTarget, such as the upper body. A configuration, i.e. a training plan consists of a set of Exercises selected to achieve the TraineeGoals and its subordinate Goals. Most Exercises require preconditions to be satisfied, for example *Equipment*, for example dumb bells, and a minimum *FitnessLevel*, e.g. *trained*, to be performed. Also, Exercises are associated with one fitness category, such as *strength*. This is because *strength* exercises cannot be used for *endurance* training. An Exercise contributes to a set of TargetGoals with (potentially) different levels of contribution, see Figure 5.
Figure 6 shows Exercise and its associations. The exercise catalog currently contains 603 exercises. For the test cases in section 5, we used different subsets ranging from 10 to 500 exercises.

Task

A Task in smartfit is the triple of TraineeGoal (TaskGoal), and Trainee- and TrainingProperties (TaskParameters) and TaskRestrictions.
Task goals: one of the TraineeGoals, selected by the user. Figure 7 shows the TraineeGoals in the knowledgebase

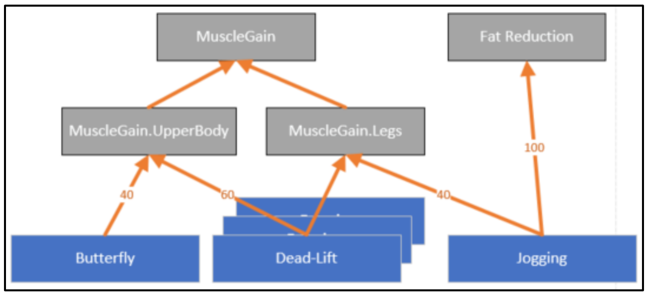

Figure 5: Component Contributions to multiple goals

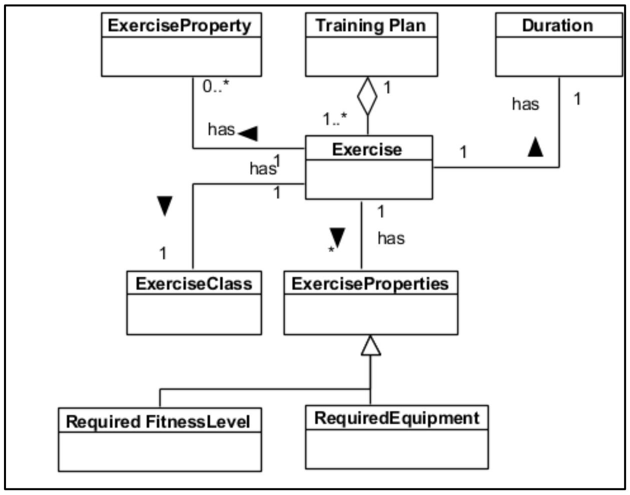

Figure 6: Exercise in smartfit

Table 2: Training and TraineeProperties

Parameter	**Values**
Age	18-40; 40-55; 55-65, 65-75; >75
Sex	Male; Female
Body-Mass-Index	<18; 18-25; 25-30; >30
Available Equipment	Machines; free weights; …
Fitness Level	Untrained, somewhat trained, trained, very trained
Working Position	Sitting, standing, overhead
Training duration	1,2,3… (given in exercises per session)
Training Focus	Body regions: upper body, core, legs

The trainee (user) is represented by a set of properties, including age, working position, body mass index (BMI) and his fitness level. The parameters and their domains are given in Table 2. Each fitness category and each body region is associated with a fitness level, e.g. *fitnesslevel(strength.upperbody)*. Initially, the trainee states either a single value after a self-assessment or performs a series of fitness tests, which in the application determine the fitness level of each category. Later, with feedback on the performed training, the specific fitness levels are refined, so that the training plans become continuously more individual. For the training, the Trainee can state a set of TrainingProperties, specifying the parameters of the training, these include currently the available equipment, the training duration (given in exercises) and the training focus. The focus allows to user to increase or decrease the priority of the active TrainingGoals and RegionGoals.

4.2 Constraints

GoalComponentConstraints

As discussed earlier, exercises are linked to the lowest level of goals, via component choices. A choice comprises all exercises that contribute to the respective goal and combines the actual contributions of the active exercises during the configuration process. A component choice is achieved if the combined contributions of all active exercises exceeds the AchievementThreshold. The threshold depends on the priority of the goal requiring the choice. Figure 3 shows the relation between Goal, Component and Choice. The domain of the contributions is currently given by $DOM(comp_i.contribution_i) = \{20,40,60,80,100\}$. The utility of a TrainingPlan in smartfit depends on the contributions of the active exercises to required Choices.

The AchievementThreshold of the Choices depends on the priority of the associated goal (24) with $DOM(Priority) = \{1,2,3,4\}$.

AchievementLevel = combine($Goal_i$.Priority, normThreshold)

The combine function for smartfit is the sum of all contributions to the choice. For each lowest level TargetGoal a choice is created. There are up to 85 TargetGoals for each TraineeGoal (42 muscle goals for strength, 42 muscle goals for flexibility, and 3 region goals for endurance).

TaskParameterGoalconstraints

There are TaskParameters, e.g. the working position, that are associated with Goals and their priorities. In the current knowledgebase, some TaskParameters can limit the priority of certain goals or exclude specific goals, i.e. prohibiting their achievement. The latter is applied for injuries or health problems, such as back pain or a broken leg. For example, for a high BMI, the priority of *strength* goals is reduced, to avoid unhealthy stress on joints or the back, and increased for endurance goals to assist *weight loss*.

So far, there are about 20 of such TaskParameterGoalConstraints in the knowledgebase. As soon as various health issues are considered, we expect this number to rise into the lower hundreds.

TaskParameterComponentConstraints

Trainee and training properties have a strong impact on exercises to be selected, mainly by excluding large sets of exercises from the component catalog. The most important examples here for are fitness level and equipment. Most exercises in the catalog do not require equipment and only a low fitness level. Thus, for a beginner with basic equipment, the majority (more than 60%) of the catalog is available. On the opposite side, for advanced exercises and special equipment, only a small subset (<20%) is available. Other examples are that certain TraineeProperties prohibit certain exercises or exercise types. E.g. a high BMI prohibits body weight exercises. The number of constraints necessary to encode this n*k were n is the number of applicable exercises and k is the number of relevant parameters. For the current knowledge base, this means that there are about 1.500 constraints. The utility for fitness training is given by Definition 6, where the cost of an exercise is given by its duration.

5 Evaluation and Case Study

To debug and assess the quality of the knowledge base and the influence of goals and parameters, we performed a set of tests with both hand-made and automatically generated instances of tasks. Besides identifying obvious bugs in the knowledge base (such as missing components in choices, improper values of contributions or priorities), the goal was to assess the adequacy of the generated training plans.

We emphasize that evaluation cannot mean checking whether smartfit generates **the** correct solution. There exists no single correct or best training plan for a task. Different human trainers will inevitably come up with different proposals. Therefore, evaluation means that experts must analyze and argue in detail whether a generated solution violates accepted principles, e.g. because it includes an inappropriate exercise or a prohibitive ordering (rather than comparing it to their own favorite plan)

This evaluation provides the feedback needed to tune parameters used in the knowledge base and to identify missing factors and constraints that influence a good training plan. The content of the domain knowledge, especially regarding the breaking down of goals, their interrelations with exercises and their quantification, is just a formal model and nothing that can be simply extracted from a textbook or guideline or would be told by a trainer. Hence, a major task now is to adjust contribution values, computation of priorities to better approximate what is judged to be a good training plan by the experts and, beyond this, to identify limitations of the chosen representation of the domain knowledge and the inferences used.

GECKO can exploit different search algorithms and constraint solvers to generate solutions. In this case study, we used haifacsp [14].

One focus of the evaluation was assessing whether the generated solutions properly reflected TaskParameters (Training- and TraineeProperties) and TraineeGoals, i.e. whether they would dedicate a reasonable amount of accumulated contributions to the various goals and sub-goals. In the following, we present the most important results and some examples.

5.1 Assessment of parameters and goal achievement

Figure 7 shows the impact of the TraineeGoals on the accumulated contributions of the configurations on the fitness categories. What Figure 7 clearly shows is that the TraineeGoals significantly influence the training plans. Strength-oriented goals, such as muscle gain or definition, have a significantly larger contribution in strength than in endurance, while more balanced goals, like general fitness show a more even distribution. Finally, weight loss and cardio contain far more endurance training than strength training. To exemplary illustrate the impact of TraineeProperties, we picked the parameter working position.

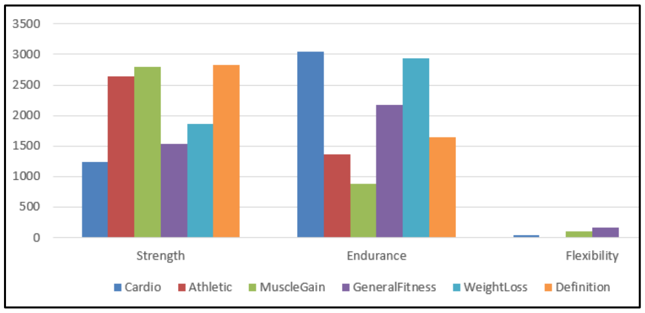

Figure 7: Comparison of Training Goal Contributions

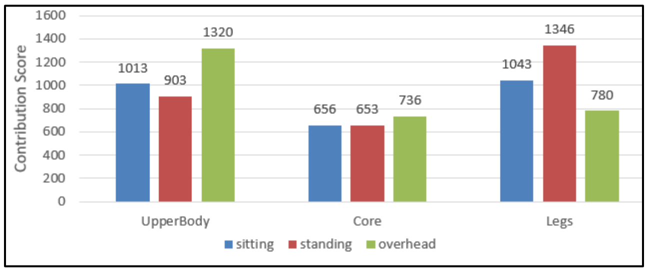

Figure 8: Working position impact

The purpose of this parameter is to support body regions that are especially stressed in a particular working position, e.g. sitting at a desk. The results are given in Figure 8. Figure 8 shows that the parameter working position changes the distribution of the exercise contributions according to the focused region. For example, for working overhead, as in construction, the focus is on the upper body, while for standing, the legs are emphasized.
Apart from technical and computational issues, it is crucial to develop a solution that users accept and that adheres to standards and practice from sport and training sciences. To evaluate the correctness and practicality of our solution, we created a set of test cases and used them to generate training plans. A series of 21 test cases was assessed by a sport scientist. Focus of the initial assessment were the usability of the training plans, with the criteria:

- Technical Correctness: are the plans correct
- Intuitiveness: are the plans understandable for trainers
- Usefulness: do the plans achieve the trainee goal
- Intensity: are the training plans appropriate for the trainee's fitness level

The majority (81%) of the training plans were correct and achieved their goals. But roughly half the training plans (54%) were too intensive for their respective trainees and a majority (63%) appeared unintuitive to the expert. Section 5.2 offers a details look at the problems and potential solutions.

5.2 Case studies

To illustrate the assessment of the training plans, we present two case studies and the conducted expert evaluation. Table 3 shows these two cases. The results of the two exemplary case studies are shown in Table 4. The use cases were chosen for detailed discussion, because one (UC1) fulfills its purpose and suffers only from minor issues, while the other (UC2) fails to fulfill its goal.
The most common faults or anomalies (ignoring bugs in the knowledge base) the expert found were:

- Training plans often contained exercises, which, seen in isolation, were correct, but in combination were too exhaustive (see Table 5: UC1)
- Training plans contained multiple versions of the same exercise, e.g. *pushup* and *pushup with narrow arms*, which is technically correct, but usually considered as faulty and inefficient by sport scientists
- Training plans with goal *cardio* are incorrect for longer training, because of the time scale (see Table 5: UC2)

6 Discussion and Future Work

GECKO has proven to be an appropriate foundation for generating training plan. Most of the evaluated test cases were considered as correct and fulfilled their purpose, but the assessment also showed some deficits.

Table 3: Case Studies

Task	**Variable**	**UC1**	**UC2**
TaskGoal	TraineeGoal	*General Fitness*	*Cardio*
TaskRestriction	Duration	*8 exercises*	*12 exercises*
TaskParameter	Age	*35*	*69*
	BMI	*normal*	*overweight*
	FitnessLevel	*Little Trained*	*Trained*
	Available-Equipment	*No Equipment*	*No Equipment*
	Working position	*sitting*	*sitting*

Table 4: Generated Training Plans

Fitness Category	**Exercises UC1**	**Exercises UC2**
Strength	Bridge_one_leg	Bridge_one_leg
	Sumo-squat	Sumo-squat
	TRX_Rollout_side	TRX_Rollout_side
Endurance	Plank with leg lifting	Push up – positive
		Back lifting
		Bridge
		Push up -single armed
		rowing
	Running	Bridge with thera-band
	Burpees	Side lifting with dumb bells
		Burpees
Flexibility	Stretching latissiumus	Stretching lower back / gluteus
	Stretching core	

Table 5: Expert evaluation of training plans

Use Case	Aspects (plan/exercise/combination of exercise)	(In-)appropriate due to	Cause	Potential solution
UC1	Training Plan	OK - Session achieves all fitness targets		
	TRX_Rollout_side	Requires exercise	Incorrect entry	Correction of ConfigKB
	Endurance exercises	The combination of burpees, plank and running is too exhaustive	Intensity of combinations not considered	
UC2	Training Plan	The training plan does not fulfill the trainee goal. Too many exercises with endurance focus, but all too short	Exercise based duration inappropriate for cardio	
	Push up positive/single arm Bridge/bridge with theraband	Two variants of the same exercise in the same plan	Both exercises have a high utility	Grouping variants in a hierarchy
	Bridge	Not an endurance exercise	Incorrect entry	s.a.

Currently GECKO does not offer a general mechanism for generating **more than one instance** of each component type, which is not a relevant restriction for a training plan, which should usually avoid repeating an exercise in the same session.

In other applications, there may be stronger **constraints on the structure** of configurations that have to be reflected during the solution generation rather than being applied a posteriori.

Prioritization of goals is the basis for another extension, which may even be relevant to smartfit: the configuration process could be iteratively related to goals with decreasing priority, thus guaranteeing that the most important goals are satisfied, even though the overall cost may not allow lower-priority goals to be fully accomplished. This also helps to break down the complexity of the task.

The current version of smartfit has an important limitation in being confined to the selection of appropriate exercises, but not fixing **how** the exercise has to be **executed**. For instance, weight-based exercises can be performed with low weight and many repetitions or vice versa and have different impacts: strength, endurance or muscle gain. Integrating the assignment of such training methods to exercises is the most important extension of smartfit.

In its current version, smartfit is designed to deliver well-designed training plans to **trainees** without detailed domain knowledge. In perspective, we want to extend smartfit to become a tool that can even support **fitness coaches** to create highly individual and complex training plans efficiently, in exploiting a knowledge base that incorporates all available state-of-the-art knowledge from training sciences.

A related potential application, with a modified knowledge base, would be planning of **physiotherapy**, following the current demand for highly personalized medical treatment. While in smartfit injuries and diseases have a restrictive impact on choosing exercises, in this context curing them would define goals that are satisfied by exercises and treatment.

Acknowledgements

We would like to thank our project partners for providing their domain knowledge and their assistance, esp. Florian Eibl from eGym GmbH. Special thanks to Oskar Dressler (OCC'M Software) for proving the constraint system (CS3) The project was funded by the German Federal Ministry of Economics and Technology under the ZIM program (KF2080209DB3).

References

[1] Kleer, J. de, Williams, B.C.: Diagnosing multiple faults. Artificial Intelligence, 32(1), pp. 97-130, (1987).
[2] Williams, B.C., Ragno, R.J.: Conflict-directed A* and its role in model-based embedded systems. Discrete Applied Mathematics, 155(12), pp. 1562-1595, (2007).
[3] Junker, U.: Configuration. In: Rossi. F., Beek, P., Walsh, T. (eds.): Handbook of constraint programming. 1st ed. Amsterdam, Boston: Elsevier, (2006).
[4] Friedrich, G.; Stumptner, M.: Consistency-Based Configuration. In: Faltings, B., Freuder, E.C., Friedrich, G., Felfernig, A. (eds.): Configuration. Papers from the AAAI Workshop. Menlo Park, California: AAAI Press (99-05), (1999).
[5] Reiter, R.: A theory of diagnosis from first principles. In Artificial Intelligence 32 (1), pp. 57–95, (1987).
[6] Kleer, J.de: An assumption based TMS. In: Artificial Intelligence 28 (2), pp. 127–162, (1986).
[7] Struss, P., 2008. Model-based Problem Solving. In: F. von Harmelen, V. Lifschitz, and B. Porter, eds. Handbook of knowledge representation. Amsterdam: Elsevier, pp. 395-465. (Struss, 2004) Struss, P.

[8] Kleer J. De, Williams BC, "Diagnosis with Behavioral Model", IJCAI, 1993.

[9] F. Grigoleit and P. Struss. Configuration as diagnosis: Generating configura tions with conflict-directed a* - an application to training plan generation. In DX@ Safeprocess, International Workshop on Principles of Diagnosis, pages 91–98. DX, 2015.

[11] MiniZinc is a free and open-source constraint modeling language. https://www.minizinc.org/
[12] SUNNY-CP: https://github.com/CP-Unibo/sunny-cp
[13] Dressler, O. and Struss, P.: Model-based Diagnosis with the Default-based Diagnostic Engine: Effective Control Strategies that Work in Practice. In: 11th European Conference on Artificial Intelligence, ECAI-94, 1994.

[15] HaifaCSP https://strichman.net.technion.ac.il/haifacsp/ Technion, Haifa

[16] Felfernig A., Hotz L., Baglay C., and Tiihonen.L.: Knowledge-based Configuration from Research to Business Cases. Amsterdam: Morgan Kaufmann, 2014