The Journal of Systems & Software 199 (2023) 111625

Contents lists available at ScienceDirect

The Journal of Systems & Software

journal homepage: www.elsevier.com/locate/jss

Goal model convergence and conflict detection for crossover services ☆

Zhengli Liu [a], Bing Li [b,*], Jian Wang [b,*], Xiangfei Lu [b], Yu Qiao [b]

[a] School of Computer Science and Artificial Intelligence, Wuhan Textile University, Wuhan, China
[b] School of Computer Science, Wuhan University, Wuhan, China

ARTICLE INFO

Article history:
Received 23 August 2022
Received in revised form 19 November 2022
Accepted 23 January 2023
Available online 6 February 2023

Keywords:
Crossover services
Requirements engineering
Goal modeling
Goal convergence
Conflict detection

ABSTRACT

As a new form of service model, crossover services aim to aggregate service resources across multiple domains to meet the complex needs of users and provide value-added services. It has received extensive attention from industry and academia due to its advantages in promoting enterprise services innovation. Crossover services span various business domains intending to meet diverse and continuously changing user requirements across multiple domains. Requirements engineering for crossover services must model user goals in various domains and converge them deeply to drive the subsequent service realization. Due to the domain heterogeneity, however, conflicts may frequently arise after the convergence of goals. Towards this issue, we propose a goal decomposition path-based method to support goal convergence of multiple domains and a computation tree logic-based method to detect conflicts between goals in the converged goal model. We evaluate the proposed method using several real cases from the literature and industry and design a controlled experiment to further assess the method's performance. Experimental results show the effectiveness of our proposed method.

1. Introduction

Faced with fiercely competitive environments and complex user needs, many companies break traditional industry boundaries and expand their business to many domains, thus providing innovative and cross-domain products and services for users. The crossover and convergence of services in multiple domains have led to the emergence of a new service form: crossover service (Chen et al., 2020). In the modern service industry, crossover and convergence are powerful ways to promote business expansion and innovation. Crossover services can create value that a single-domain service cannot create, thus achieving the value emergence effect of "1 + 1 > 2" (Guo et al., 2019b). Crossover is a process by which a company introduces new services or upgrades existing services to realize value creation or innovation. Enterprises improve resource utilization efficiency and promote the collaboration of organizations in different domains through the crossover. Convergence further promotes the exchange and sharing of data, information, knowledge, and value among various enterprises (Shan et al., 2020). Crossover and convergence are becoming increasingly popular, and crossover services have become a new trend in the modern services industry. For example, Amazon, a new retail and cloud computing service provider, has entered the medical domain and is now offering healthcare services.

Crossover services typically involve multiple business domains, and user requirements can be personalized and diverse. How to capture and converge users' objectives in different domains is an essential topic in the crossover service requirements analysis process. Among various requirements analysis methods, goal-based Requirements Engineering (RE) has been widely adopted due to its capabilities to capture the intentions of stakeholders and provide a good understanding of the systems to be developed (Ali et al., 2010). Simultaneously, goal-based modeling methods can further elicit large-grained goals through AND/OR decompositions to obtain fine-grained sub-goals (Qian et al., 2018). In addition, goal-based modeling methods can identify potential conflicts by analyzing constraint relations between goals like dependence and exclusion. Therefore, researchers have adopted goal-based methods to deal with user requirements analysis of crossover services (Li et al., 2020; Liu et al., 2020; Peng et al., 2020). The purpose of requirements convergence analysis does not mean abandoning the original requirements models but reconstructing and improving the original requirements models in light of evolving crossover requirements (Shan et al., 2020). However, existing goal-based methods do not address the requirements convergence problem for crossover services. Although significant achievements have been made in crossover services modeling, there is still no systematic approach for resolving the requirements convergence and conflict detection issues of crossover services.

☆ Editor: Xiao Liu.

* Corresponding authors.
E-mail addresses: lzl@wtu.edu.cn (Z. Liu), bingli@whu.edu.cn (B. Li), jianwang@whu.edu.cn (J. Wang), xiangfeilu@whu.edu.cn (X. Lu), qiaoyu_cs@whu.edu.cn (Y. Qiao).

https://doi.org/10.1016/j.jss.2023.111625
0164-1212/

With the evolving needs of users, new goals may arise that are hard to be achieved in the current domain. We refer to these new goals as to-be-converged goals and the domain as a subject domain. Since the to-be-converged goals cannot be satisfied in the subject domain, we must seek solutions in other domains that enable them to be satisfied. Object domains refer to these domains that can supply necessary solutions. Since service convergence involves multiple domains and is an important aspect of crossover services, determining how to converge goal models from different domains thus becomes a key research problem for crossover services (Shan et al., 2020). Converging user goals from other domains provides a feasible way to meet the complex requirements of users. However, since goal constraints in different business domains are usually different, conflicts will inevitably arise in the convergence process. For example, in a typical crossover service case, Fliggy,[1] a functional goal of a hotel provider is "Selling rooms on an online platform", and a functional goal in the insurance domain is "Buying insurance offline". When the two goals are converged, there will be a conflict because Fliggy does not support the offline purchase model. Therefore, it is necessary to detect conflicts between goals to ensure the consistency of the converged goal model.

Despite the success of goal-oriented approaches in requirements engineering, there are limitations in applying them in the requirements convergence analysis for crossover services. Firstly, these classical goal-based modeling methods are mainly designed to model user goals in a single domain, which cannot directly achieve the convergence of user goals in multiple domains. Secondly, they are unable to detect potential conflicts during the convergence process.

Towards these issues, we propose a goal convergence method based on goal decomposition paths. Moreover, a Computation Tree Logic (CTL) based method is proposed to detect conflicts between goals. In the experiments, we use Fliggy, a real-world case, to validate the proposed goal convergence method and additional real-world cases to evaluate the goal conflict detection method. Three research questions (RQs), which serve as the impetus for this research, are investigated simultaneously.

RQ 1: Can our proposed method realize the convergence of goal models from different domains in crossover scenarios?

RQ 2: Can our proposed method identify goal conflicts?

RQ 3: Is our proposed method easy to understand and use by requirements engineers?

In particular, the main contributions of this paper are summarized as follows:

(1) We propose a goal convergence method based on goal decomposition paths, which supports the convergence of goal models in multiple domains.

(2) We propose a method for detecting goal conflicts in converged goal models.

(3) We develop a prototype to support the goal model convergence in a visualized way and the conflicts detection of the converged goal model.

The rest of the paper is organized as follows. In Section 2, we introduce the related works and give an illustration example to illustrate the motivation of this work. Section 3 explains the preliminaries of our proposed method. Section 4 introduces the details of our proposed goal convergence method. In Section 5, we illustrate the details of our proposed goal conflict detection method. Section 6 introduces our developed tool. In Section 7, we conduct a set of experiments to evaluate our proposed method and discuss evaluation results. Finally, Section 8 concludes our study and puts forward future work.

[1] https://www.fliggy.com.

2. Related work and an illustration example

In this section, we summarize the related work on crossover services RE. First, we present existing works from three perspectives: goal-oriented RE, goal conflicts detection, and crossover services modeling. Second, we compare how our method differs from the considered works. Third, we use an example to illustrate the motivation of this work.

2.1. Goal-oriented requirements engineering

Goal models have attracted the attention of many researchers. Liu et al. proposed a value-driven modeling method for crossover service RE. The proposed method supports the modeling and analysis of crossover scenes from the perspectives of value, goals, and services (Liu et al., 2020). Bunyadi et al. proposed an integrated method that combines the KAOS model and the Six-Variable model to support the decomposition of soft goals (Ulfat-Bunyadi et al., 2019). Nguyen et al. proposed an extended goal modeling language that can express preferences between goals, assign numerical attributes to goals, and define constraints and optimization goals over multiple objective functions (Nguyen et al., 2018). Malak et al. proposed a goal integration method for different initial contexts, which realizes the completeness and consistency analysis of the integrated goal model and provides traceability to rationales and decisions made during integration (Baslyman and Amyot, 2019). According to the specification of a certain relation between goals, the proposed methods (Giorgini et al., 2002; Letier and Van Lamsweerde, 2004; Giorgini et al., 2005) mainly focus on if a refinement of a goal model is a consistent way to accomplish particular objectives. In addition, only a few works (Peng et al., 2020) study the goal convergence problem in crossover scenarios. Our work enriches these works in analyzing the goal convergence of multiple domains. We explore the problems faced in the requirements convergence analysis of crossover service and propose a method to support the convergence of goal models in different domains. Aiming at the potential conflicts that may appear in the converged goal model, a CTL-based method is presented to identify the potential conflicts and ensure the consistency of the converged goal model.

2.2. Goal conflicts detection

In terms of goal conflict detection, several works studied this problem and proposed some detection methods. Ali et al. studied the contextual goal model, including the variability of both context and goals. They used an SAT-based method to detect and analyze inconsistency and conflicts in a goal model (Ali et al., 2013). Asadi et al. proposed a description logic-based method to represent goal models and their relations. They employed a reasoning method to detect inconsistencies in the goal and feature models (Asadi et al., 2016). Degiovanni et al. utilized model counting technologies to evaluate the probability of goal conflicts and how these affect goal satisfaction (Degiovanni et al., 2018a). Murukannaiah et al. proposed a novel approach Arg-ACH to capture inconsistencies between stakeholders' goals and beliefs and resolve goal conflicts (Murukannaiah et al., 2015). Chatzikonstantinou et al. introduced a framework to support run-time reasoning for medium and large-scale fuzzy goal models (Chatzikonstantinou and Kontogiannis, 2018). Degiovanni et al. proposed a genetic-based method to identify goal conflicts. They used an LTL satisfiability checker to guide the genetic algorithm to search for formulas that capture divergences in the specification (Degiovanni et al., 2018b). In addition, they also proposed a linear-time temporal logic-based method to detect goal conflicts (Degiovanni et al., 2016). Jorge et al. implemented Badger, a regression

planner in Prolog that generates resolution plans to resolve inconsistencies of design models in model-driven software development (Pinna Puissant et al., 2013). Existing research focuses on checking the consistency of individual goal models in a single domain, but cross-domain goal convergence lacks systematic research. Moreover, the existing methods also face deficiencies in detecting goal conflicts. Our work alleviates this problem by providing reasoning mechanisms to detect whether a conflict happens. This can help requirements analysts find modeling errors in the early stages of requirements analysis and avoid passing to the software service design and development stage.

2.3. Crossover services modeling

Yin et al. defined that crossover services involve the deep convergence of pattern, ecosystem, environment, quality, and value (Yin et al., 2018). Guo et al. proposed an interactive fusion method for crossover health services based on microservice architecture. Driven by the service fusion requirements, their method detects the business inconsistency between the services to be fused and realizes business process reengineering through human–computer interaction. Then, the semantic inconsistency in service interface matching is detected and solved by splitting and perfecting the parameter concept. Complex business services are converged based on the microservice architecture (Guo et al., 2019a). Xi et al. proposed a scenario-based requirements engineering method SBRM for crossover healthcare service modeling. Their proposed method focuses on scenarios, processes, and rules, in what scenario, which process must be executed, and which rule must be satisfied (Xi et al., 2019). Liu et al. proposed a value-driven meta-model for crossover services modeling, which includes a value network, a goal network, and a service network. The proposed method can support the requirements analysis of crossover services from multiple viewpoints (Liu et al., 2020). Chen et al. proposed a temporal–spatial-domain ternary model to define some value-related factors (KPIs about activities, processes and collaborations, constraints on activities and resource utilization) on standard BPMN 2.0 processes, making this information be evaluated, traced, and improved in applications (Chen et al., 2020). Xue et al. proposed a value-based analysis framework for crossover services from the perspective of value creation and realization, and the proposed method includes four parts: value assessment, data analysis, capability analysis, and execution path (Xue et al., 2021). Shan et al. proposed a message flow partition and merging-based process convergence approach with consideration of involved roles and goals (Shan et al., 2020). Peng et al. proposed a role-based goal convergence and conflicts detection method and a TF-IDF-based conflicts resolution method to solve the possible conflicts in the converged goal model (Peng et al., 2020). Li et al. proposed a collaborative modeling approach for crossover services. The method divides the modeling process into a pattern layer and an instance layer according to the global and local level dependencies of crossover services collaboration. A mapping relationship between the global service orchestration process and the local business execution process is established to model the hierarchical constraints of the abstract decision layers and concrete implementation layer (Li et al., 2020). Although some methods have been proposed for crossover services modeling, there is still a lack of requirements convergence and conflict detection mechanisms to support crossover scenarios. We conduct a series of exploratory and formative studies and put forward a goal model convergence and conflict detection method for crossover services to ensure the consistency of the converged goal model.

Table 1
Comparison of different methods.

Methods	Criteria			
	Requirements modeling	Requirements convergence	Conflicts detection	Tool support
Chen et al. (2020)	+	–	–	–
Shan et al. (2020)	–	+	–	–
Liu et al. (2020)	+	–	–	+
Peng et al. (2020)	–	+	+	+
Ulfat-Bunyadi et al. (2019)	–	–	–	–
Nguyen et al. (2018)	–	–	+	+
Baslyman and Amyot (2019)	–	–	+	+
Giorgini et al. (2002)	–	–	–	–
Letier and Van Lamsweerde (2004)	–	–	–	–
Giorgini et al. (2005)	–	–	+	+
Ali et al. (2013)	–	–	+	+
Asadi et al. (2016)	–	–	+	+
Degiovanni et al. (2018a)	–	–	+	–
Murukannaiah et al. (2015)	–	–	+	–
Chatzikonstantinou and Kontogiannis (2018)	–	–	+	–
Degiovanni et al. (2018b)	–	–	+	–
Degiovanni et al. (2016)	–	–	+	–
Pinna Puissant et al. (2013)	–	–	+	–
Guo et al. (2019a)	–	+	–	–
Xi et al. (2019)	+	–	–	–
Xue et al. (2021)	+	–	–	+
Our method	+	+	+	+

2.4. Comparison of methods

Here we compare our proposed method with other related works based on predefined criteria, and the results are shown in Table 1. Table 1 shows the difference between our method and related work according to the predefined criteria. Our work focuses on goal model convergence and goal conflict detection. We proposed a decomposition path-based method to achieve the convergence of goal models of different domains and a computation tree logic-based method to detect conflicts between goals, thus ensuring the consistency of the converged goal model. However, most of the related works only consider requirements modeling. Requirements analysis for crossover services, such as goal convergence and goal conflicts detection, are ignored.

2.5. Illustration example

The problem to be addressed in this paper is illustrated using the following example. Fliggy is an OTA (Online Travel Agency) platform whose original goal is to provide hotel booking services for users. With the evolution of user needs, a user may want to reduce the potential loss caused by unsubscription, which cannot be satisfied in the current domain (i.e., the subject domain). Fortunately, the insurance domain (i.e., an object domain in this example) provides such solutions that can make the unsatisfied goal satisfied. Therefore, the goal models of the subject domain and the object domain should be converged. Figs. 1 and 2 are the goal models of the OTA domain and the insurance domain, respectively. The to-be-converged goal in the subject domain and the solutions in the object domain are marked with a red dotted box. Our objective is to converge these models together, thus making the goal of “Buy insurance” satisfied.

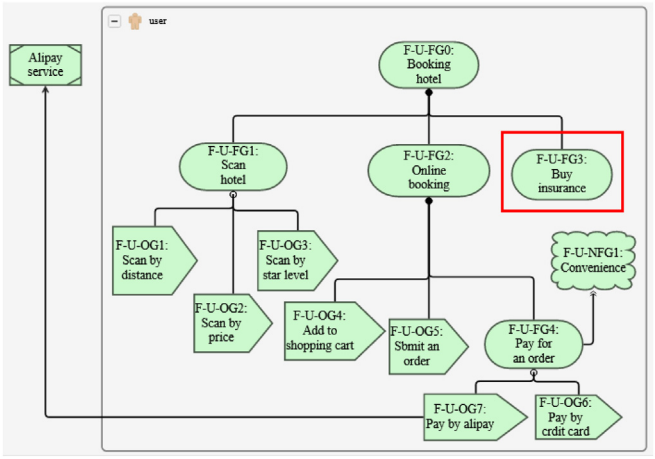

Fig. 1. User goal model in the OTA domain.

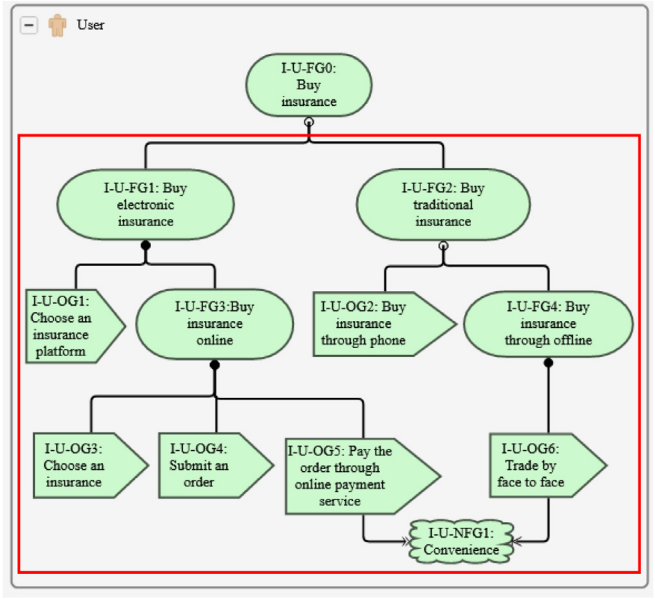

Fig. 2. User goal model in the insurance domain.

In general, the two goal models are consistent in their respective domains; however, conflicts may occur after the two goal models are converged because of the heterogeneity of the two domains. Therefore, we need to check whether the converged goal model is consistent.

3. Preliminaries

In this section, we introduce some preliminaries related to our proposed method.

3.1. Computation temporal logic

CTL is a kind of propositional branching temporal logic used to describe sequential relations and branches of system states. Given a set of propositional variables, *AP*, CTL formulas are defined using standard logical connectives and temporal operators, as follows: (1) propositional variable p $\in$ AP is a CTL formula, (2) if f and g are CTL formulas, so are $\neg$f, f $\vee$ g, f $\wedge$ g, **A** $\Box$ f, **E** $\Box$ f, **A** $\Diamond$ f, **E** $\Diamond$ f, **A** $\circ$ f, **E** $\circ$ f, **A**($f \triangleright g$), and **E**($f \triangleright g$). Here we explain the temporal operators mentioned above. $\Box$ is an always operator,

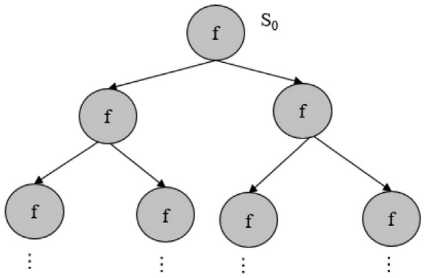

Fig. 3. Example of $A \Box f$.

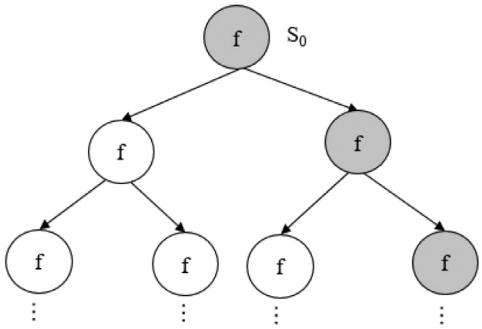

Fig. 4. Example of $E \Box f$.

$\Diamond$ is a sometimes operator, $\circ$ is a next operator, and $\triangleright$ is an until operator. **A** (for all paths) and **E** (there exists a path) are path quantifiers. A CTL formula consists of path quantifiers and temporal formulas.

A semantic model of CTL is a Kripke (Emerson and Clarke, 1982) triple $M = (S, R, L)$, where $S = \{s_0, s_1, s_2, \ldots, s_n\}$ is a non-empty state set, $R \subseteq S \times S$ is a binary relation defined on S, which represents a transfer relation between all possible states, i.e. $\forall s_i \in S$, $\exists s_j \in S$, $(s_i, s_j) \in R$; and L is a label function to return the set of all atomic propositions in S that are true. A path of Kripke is a finite sequence of states in S, denoted as $\pi = s_0, s_1, s_2, \ldots, s_n$, and $\forall\, i \geq 0$, $(s_i, s_{i+1}) \in R$, where π^i represents the suffix of the path starting from s_i, i.e. $\pi^i = s_i \rightarrow s_{i+1} \rightarrow s_{i+2} \rightarrow \cdots \rightarrow s_n$. Note that there may be many uncertain states in the future of the CTL semantic model, i.e. from the perspective of the state path, each state may branch to multiple states. An intuitive explanation for CTL semantics is shown as follows. Assume that s_0 is a state of M in a Kripke triple, and f and g are two CTL formulas. Figs. 3 and 4 explain the following CTL semantics.

$$\mathrm{M},\ s_0 \models A \Box f \ \textit{iff}\ \forall\, \pi = s_0 \rightarrow s_1 \cdots \rightarrow s_n,\ \forall\, i \geq 0,\ M,\ s_i \models f; \quad (1)$$

$$\mathrm{M},\ s_0 \models E \Box f \ \textit{iff}\ \exists\, \pi = s_0 \rightarrow s_1 \cdots \rightarrow s_n,\ \forall\, i \geq 0,\ M,\ s_i \models f; \quad (2)$$

3.2. Binary decision diagram

Binary decision diagrams (BDDs) (Drechsler and Sieling, 2001) provide a convenient way to represent and manipulate Boolean functions in a symbolic form. They are particularly effective as the algorithmic basis of symbolic model checkers. A BDD represents a Boolean function as a directed acyclic graph with only one root node, denoted as $G = (V, E)$, where the node set V contains two node types: nonterminal vertices and leaf vertices. For nonterminal vertex v, its labeled variable is denoted as $var(v)$. In contrast, $val(v)$ is used to represent the associated value of a leaf vertex v. For each nonterminal vertex $v \in V$, it has two child vertex types, represented as $high(v)$ and $low(v)$, corresponding to the case that

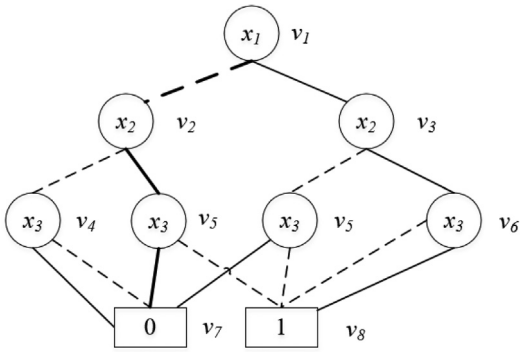

Fig. 5. BDD of Boolean function f.

its variable value is 1 and 0, respectively. The two leaves are also called l-*leaf* and 0-*leaf*. For example, given a Boolean function $f = x_1\overline{x_3} + x_1x_2 + x_2\overline{x_3}$, its BBDs are shown in Fig. 5, where a nonterminal vertex is denoted by a cycle, and a terminal vertex is denoted by a rectangle. The dashed line represents 0-leaf, and the solid line represents 1-leaf. The meaning of the path marked with a thick line in the figure is that when x_1, x_2, and x_3 are assigned with the value 0, 1, and 0, the value of function f is 0.

We can associate f_v with each vertex v in the graph to define the Boolean function. For terminal vertex v, the following associated function is satisfied.

$$\begin{cases} if\ var\ (v) = 1,\ then f_v = 1 \\ if\ var\ (v) = 0,\ then\ f_v = 0 \end{cases} \tag{3}$$

For nonterminal vertex v, the associated function can be defined as follows.

$$f_v = var\ (v) \cdot f_{high(v)} + \overline{var(v)} \cdot f_{low(v)} \tag{4}$$

Every vertex in a BDD represents a Boolean function. The two children of a vertex correspond to its two cofactors concerning its associated variable.

With ordered binary decision diagrams (OBDDs) (Wang, 2004), an ordering rule is added to the graph vertex variables. Given a BDD $G = (V, E)$, we enforce a total order relationship $\prec$ on a set of variables. For any nonterminal vertex $u \in$ V, if nonterminal vertex $v \in$ V and $u \in \{low(v), high(v)\}$, then $var(u) \prec var(v)$. In Fig. 5, we can see that the variable indices along all paths from the root to leaves are in increasing order, and thus it is an OBDD.

For an OBDD, if it satisfies the following rules, it is a reduced OBDD (i.e., ROBDD), a canonical form of a Boolean function.

(1) There can be, at most, one leaf having a given value.
(2) For any nonterminal vertex v, $low(v) \neq high(v)$ must hold.
(3) For any $v, v' \in V$ and $v \neq v'$, the following conditions $var(v) \neq var(v')$, $high(v) \neq high(v')$, and $low(v) \neq low(v')$ must be satisfied.

Given an arbitrary OBDD, we can use the following rules to convert it into ROBDD in linear time complexity.

(1) Rule 1 (**Elimination rule**): For a vertex v in an OBDD, if $low\ (v) = high\ (v)$, then eliminate vertex v and redirect all incoming edges to its child.
(2) Rule 2 (**Merging rule**): For two vertices v and v', if $var\ (v) = var\ (v')$, $low\ (v) = low\ (v')$ and $high\ (v) = high\ (v')$, then eliminate one of the vertices and redirect all incoming edges to the other one.
(3) Rule 3 (**De-duplication rule**): For two leaves, u and v, if $val\ (u) = val\ (v)$, then remove any one of them and redirect all incoming edges to the other.

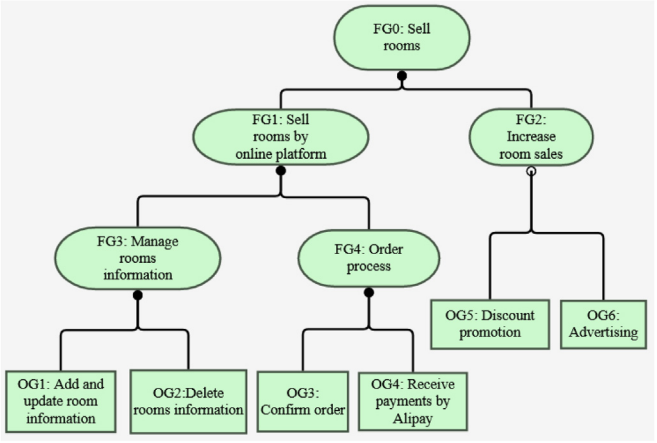

Fig. 6. An example of a goal tree model.

More details about BDD can refer to Clarke et al. (2018).

4. Method of goal convergence

This section introduces the detail of our proposed goal convergence method. We first introduce some definitions of our proposed method. Second, we present the details of the proposed goal convergence algorithm.

4.1. Definitions

For simplicity, we use a tree structure (also known as a goal tree) to represent a goal model. A goal tree contains multiple types of nodes and links. An example is given in Fig. 6. In what follows, we will introduce some important concepts of the goal tree.

Definition 1 (*Goal Tree*)**.** A goal tree T $= (\upsilon, \varepsilon)$ with a node type mapping $\emptyset : \upsilon \rightarrow$ G and a relation mapping function: $\psi : \varepsilon \rightarrow \mathcal{R}$, where υ, ε, G, and $\mathcal{R}$ denote a node set, a relation set, a node type set, and a relation type set, respectively. The tree schema for the goal tree T denoted as $Z_T = (G, \mathcal{R})$, is a tree whose nodes are node types from G and links are relation types from $\mathcal{R}$. We refer to a goal that has only lower-level goals but no upper-level goals as a root goal, and we refer to goals that have only upper-level goals but no lower-level goals as leaf goals.

Definition 2 (*Decomposition Path*)**.** A decomposition path P is defined on the tree schema $Z_T = (G, \mathcal{R})$, with the form of: $G_1 \xrightarrow{R_1} G_2 \xrightarrow{R_2} G_3 \xrightarrow{R_3} \cdots \xrightarrow{R_l} G_{l+1}$, which defines a composition relation of $R = R_1 \circ R_2 \circ R_3 \cdots R_l$ between types G_1 and G_{l+1}, where $\circ$ represents the composition operator between relations. Note that a goal decomposition path must end with a leaf goal. A typical example of the decomposition path is "FG_1: Sell rooms by online platform $\xrightarrow{AND}$ FG_3: Manage room information $\xrightarrow{AND}$ OG_1: Add and update room information".

Given that there are rich relations in a goal model, a matrix for a goal decomposition path P is defined as M_P, where $M_P\ (i, j)$ represents relations between G_i and G_j, i.e. $M_P\ (i, j) = R_{G_iG_j}$. Note that there are two kinds of relations in a goal model. One is the decomposition relation between the high-level goals and the lower-level goals, including "AND" and "OR"; the other is the constraint relations between goals, including obstacles, dependence, exclusion, and contribution. For ease of representation and

Table 2
The relations between goals and their codes.

No.	Relation	Remark
1	AND (G_a, {G_b})	Upper goal G_a can be achieved only if all the lower sub-goal set G_b associated with it are achieved.
2	OR (G_a, G_b)	The upper goal can be achieved if any one of the lower sub-goal associated with it is achieved.
3	Obstacle (G_a, G_b)	Goal G_a provides sufficient evidence against the satisfaction of goal G_b.
4	Dependence (G_a, G_b)	The realization of a goal G_a depends on another one G_b.
5	Exclusion (G_a, G_b)	Two goals, G_a and G_b, are incompatible with each other; that is, the realization of goal G_a will lead to the failure of another goal G_b.
6	Promotion (G_a, G_b)	A goal G_a provides positive evidence for the satisfaction of another goal G_b.

storage, we use numbers 1–6 to refer to these relations. The name of the relations and their codes are shown in Table 2. If there is no relation between G_i and G_j, $M_P(i, j) = 0$. To distinguish the direction of the relations between goals, we use codes and their opposite number to represent different relations. For example, in the goal model shown in Fig. 6, goal FG_3 is a subgoal of goal $FG1$, and the relation between them is "AND". Thus $M_P(FG_3, FG_1) = 1$ and $M_P(FG_1, FG_3) = -1$.

4.2. Decomposition path based goal convergence

For goal convergence, a key task is to find an appropriate convergence point from the goal model of the object domain according to semantic similarity. A convergence point is a goal node, indicating where to extract the decomposition path. The accuracy of the matching result is the premise of accurate convergence of goal models in different domains. Note that a goal is usually expressed by a phrase or sentence, such as "provide protection for customers with uncertain itinerary and reduce the loss caused by cancelling orders". Therefore, we need to measure the goal similarity from the sentence level. Based on the above considerations, we use a start-of-the-art method Sentence-BERT (Reimers and Gurevych, 2019), to convert our goal description into a vector and then use the cosine similarity to calculate the semantic similarity of two goals from different domains.

$$\cos(G_1, G_2) = \frac{\sum_{i=1}^{n}(G_{1i} \times G_{2i})}{\sqrt{\sum_{i=1}^{n} G_{1i}} \times \sqrt{\sum_{i=1}^{n} G_{2i}}} \tag{5}$$

If the similarity between two goals is greater than a threshold α, the two goals will be considered to be similar and thus be a potentially converged pair. When we find a goal model from object domains that can make the to-be-converged goal of the subject domain satisfied, the next step is to converge the two goal models. To address this problem, we propose a goal convergence method based on the decomposition path, which decomposes goal fragments into paths and achieves goal convergence based on paths. To extract the decomposition path efficiently and accurately from the goal model, we set the convergence point as the starting vertex and use a depth-first traversal algorithm to obtain a goal decomposition sequence.

Consider that there are decomposition relations and constraint relations between goals, and these relations need to be considered in the goal convergence process. Therefore, we use a relation matrix M_p to store the relations of goals, and these relations are represented by the codes defined in Table 2. An example can be seen in Eq. (6).

$$M_p = \begin{bmatrix} 1 & \cdots & 0 \\ 0 & \cdots & 0 \\ 2 & \cdots & 2 \end{bmatrix} \tag{6}$$

After we obtain the useful decomposition paths, we can execute the goal convergence task. The details of our convergence method are described below.

Algorithm 1: Goal convergence algorithm (GC)
Input: Two goal models of different domains
Output: Converged goal model
1: C_p←∅;
2: similarity, convergePoint =Sim (tcgoal, objectModel) // Find the convergence point
3: **if** similarity>α **then**
4: C_p = DFS (convergePoint, objectModel); // Find the convergence point
5: **for** i←0 to length(C_p) **do**
6: convergedGoalModel = Convergence (C_p[i]) // Traverse C_p and converge it into the subject goal model
7: annotateRelation (convergedGoalModel, M_p) // Annotate relations between goals
8: **else**
9: **return** false

Given two goal models from different domains that need to be converged, we first use the similarity algorithm to find a similar goal from the object domain. Then the depth-First-search(DFS) (Tarjan, 1972) algorithm is used to extract decomposition paths. After that, the decomposition path C_p will be converged into the goal model of the subject domain. When goal convergence is completed, the next task is to annotate relations for the converged goals. According to the relation matrix M_P, the relations between goals of the converged goal model will be annotated. Considering that the goals of the object domain need to rely on some resources and goals of other roles, we need to converge these resources and goals into the subject domain.

In our algorithm, we assume that goal models of the subject domain and the object domain exist, for example, in a historical project document or a database. However, in some cases, the goal model of the object domain may not exist. In this case, the requirements engineers need to use an object-oriented modeling language to build the goal model, such as *i** (Dalpiaz et al., 2016).

5. Conflict detection of goal convergence

It is worth noting that there are multiple types of conflicts between goals. This paper examines the detection of goal conflicts that arise during requirements convergence. In their own domains, goals are consistent, but when they are converged, conflicts arise due to the heterogeneity and constraints of the domains. To check the conflicts between goals, we need to transfer the goals specified by words or sentences into CTL formulas. Besides Atomic Proposition (AP), constant true, and Boolean connectives ($\neg$, $\bigvee$), the following temporal operators are sufficient: **E X**, **E G**, **E**[**U**]. Without loss of generality, this paper restricts the syntax of CTL as follows:

$$\varphi = \mathrm{p} \mid \neg\varphi 1 \mid \varphi 1 \wedge \varphi 2 \mid EX_{\varphi 1} \mid EG_{\varphi 1} \mid E(\varphi 1 U \varphi 2) \tag{7}$$

Algorithm 2 illustrates the details of our proposed detection procedure. This work mainly focuses on the CTL formulas like (7). Note that other operators can be transferred using stand equivalences (C. Edmund et al., 2000; Clarke et al., 2018). Given the goals represented by CTL formulas, their conjunction will be used as the input of our algorithm. The details of the detection algorithm are shown in Algorithm 2.

Algorithm 2 Goal conflict detection method
Input: a set of goals g_c, and the goals are represented by CTL specifications
Output: If there is no conflict, the procedure returns True; otherwise, the procedure returns False and outputs the unsatisfiable formula and goal number.
1: **function** detectGoalConflict (M, s_0, g_c)
2: for i = 1 to g_c.length
3: for j =1 to g_c.length do
4: $\varphi = g_c^i \wedge g_c^j$
5: S= Check (M, φ)
6: **return** ($s_0 \wedge \neg S$)=0
7: **function** Check (M, φ)
8: **if** $\varphi = p$, **then return** p
9: **else if** $\varphi = \neg\varphi 1$ **then return** ¬ Check (M, $\varphi 1$)
10: **else if** $\varphi = \varphi 1 \wedge \varphi 2$ **then**
11: **return** Check(M, φ1) ∧ Check(M, φ2)
12: **else if** $\varphi = EX\varphi_1$ **then**
13: S_φ=Check (M, φ1))
14: **return** predecs (M, S_φ)
15: **else if** $\varphi = EG\varphi_1$ **then**
16: **return** EGCheck(M, Check(M, φ1))
17: **else if** $\varphi = E(\varphi_1 U \varphi_2)$ then
18: **return** EUCheck(M, Check(M, φ1), Check(M, φ2))
19: **end function**
20: **function** EGCheck(M, S_φ)
21: $S' := S_\varphi$
22: **while** $S' \neq S$ **do**
23: S: = S'
24: $S' := S_\varphi \wedge predecs(S, R)$
25: **end while**
26: **return** S
27: **end function**
28: **function** EUCheck(M, S1, S2)
29: $S' := \emptyset$
30: **while** $S' \neq S$ **do**
31: S := S'
32: $S' := S_2 \vee (S_1 \wedge predecs(S, R))$
33: **end while**
34: **return** S
35: **end function**

In Lines 1–6, we first construct the conjunction of the goals and then check if the formulas can be satisfied to identify whether the goal is consistent. In Lines 7–19, the function *check* checks if a conflict occurs, and it uses different functions to check the CTL formulas based on the parameter φ. Specifically, it uses functions *EGCheck* and *EUCheck* to address the case where φ is of the form $EG_{\varphi 1}$, E ($\varphi 1 U \varphi 2$), respectively. In Line 14, the function *predecs* is used to obtain the predecessors of S_φ. The predecessors are the set of all states that can move one step to reach S_φ according to the transition relation R. In Lines 20–27, the function *EGCheck* uses a greatest fixed point computation to iteratively compute a set of states in which there is a path composed only of the states of S_φ. S_i is the value of S in the ith iteration. Initially, S_0 contains all states. Intuitively, the *EGCheck* algorithm assumes that all states in S have a successor. In the first iteration, we assign the set of states in S_0 with at least one immediate successor to S_1 and delete the states without successors in S_0. Following this reasoning, in iteration i, S_i contains all states with a path of length $i-1$ in S_{i-1}. When the algorithm terminates, S contains only states with an infinite path contained in S. In the continuous iteration, S is continuously reduced. Since it is finite initially, the algorithm eventually terminates. In Lines 28–35, the function *EUCheck* uses the least fixed-point computation to compute the set of states from which a path exists on which states from S_1 appear initially until a state from S_2 appears. In the algorithm, we use a strong version of the until-operator E (φ_1 U φ_2), which requires φ_2 to hold in the last state of the satisfiable computation.

Table 3
Propositions of the insurance domain.

Proposition ID	Goal ID	Goal description
q_1	$I\text{-}U\text{-}OG_1$	Choose an insurance platform
q_2	$I\text{-}U\text{-}OG_2$	Buy insurance through phone
q_3	$I\text{-}U\text{-}OG_3$	Choose an insurance
q_4	$I\text{-}U\text{-}OG_4$	Submit an order
q_5	$I\text{-}U\text{-}OG_5$	Pay the order through online payment services
q_6	$I\text{-}U\text{-}OG_6$	Trade by face to face

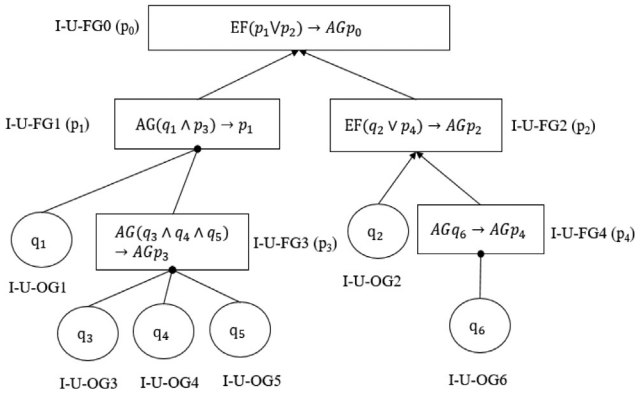

Fig. 7. User goal model represented by CTL formulas.

We propose automatically detecting such conflicts by representing and processing CTL formulas from goals' temporal formalizations. Let us use an example to illustrate how the proposed methods work.

For the case of the Fliggy crossover service, we first represent the goals of the converged goal model with CTL formulas. Here we use the user goal model of the insurance domain as an example, as shown in Fig. 2. We traverse the goal model and extract the operational goals. The obtained operational goal list is shown in Table 3.

These operational goals will be defined as an atomic proposition to represent the upper functional goals. The goal model represented by CTL formulas is shown in Fig. 7.

From the above Fliggy crossover service case, some of the functional goals represented by CTL formulas are given below. Requirements engineers can provide these formulas according

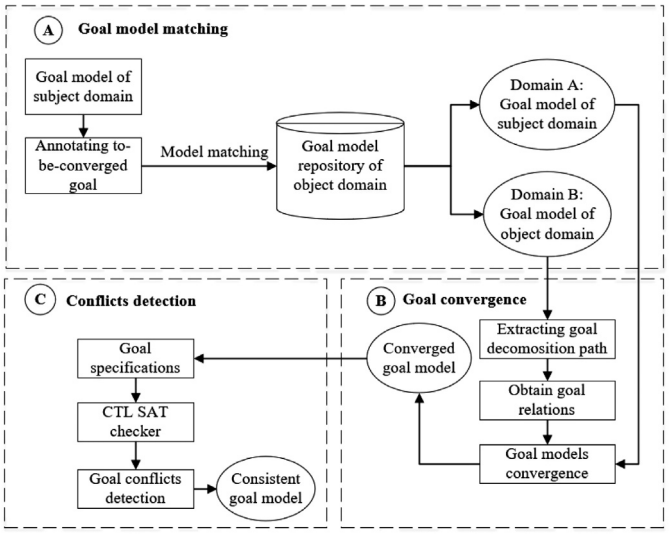

Fig. 8. Detection process of our approach.

to their knowledge and skills. Our proposed method will detect conflicts between goals when these formulas are given.

Goal: I-U-FG1 (p_1) Buy electronic insurance

Formal Definition: $AG(q_1 \wedge p_3) \rightarrow p_1$

Goal: I-U-FG2 (p_2) Buy traditional insurance

Formal Definition: $\mathrm{EF}(q_2 \vee p_4) \rightarrow AGp_2$

Goal: I-U-FG3 (p_3) Buy insurance online

Formal Definition: $\mathrm{AG}(q_3 \wedge q_4 \wedge q_5) \rightarrow AGp_3$

Goal: I-U-FG4 (p_4) Buy insurance offline

Formal Definition: $AGq_6 \rightarrow AGp_4$

To check whether the converged goal model is consistent, we construct the conjunction of two functional goals to test whether the formula can be satisfied. For example, to check whether the functional goal I-U-FG2 of the insurance domain is consistent with the goal F-F-FG3 of the Fliggy domain, the formula "$AGq_{18} \wedge \mathrm{AG}(q_{19} \wedge q_{20} \wedge \neg(q_{15} \wedge q_{16} \wedge q_{15} \wedge q_{16}))$" is constructed and treated as an input of our proposed detection method. The details of the detection method are shown in Algorithm 2.

6. Tool

To support the practice of the proposed method, we have developed a prototype system called the Goal Convergence tool (GCon). The steps of goal convergence and conflict detection are shown in Fig. 8.

In addition, some interfaces of our tool are shown in Figs. 9, 10, 11, and 12. This tool has been developed to demonstrate the usefulness of our proposed method when applied in practice. The usage of the tool includes four steps:

(1) Import goal models. Requirements engineers should first import the goal model of the subject domain into the tool if the goal model exists. However, if the goal model does not exist, requirements engineers must construct it from scratch, which can also be supported by our tool.

(2) Converge goal models. After the goal model of the subject domain is imported, our proposed method searches the most similar goal to the to-be-converged goal from the database and imports it into the goal convergence interface of the tool. The import result can be seen in Fig. 9. The right side of the design interface is the goal model found from the database. After the two goal models are imported, our tool can assist the requirement engineer in completing goal model convergence based on the proposed method.

(3) Annotate goal relations. After the to-be-converged goal models have been converged, the relations between goals need to be annotated in this step. Requirements engineers need to transform the functional goals into CTL specifications so that our proposed method can identify if there exists a conflict. Our tool can generate CTL formulas according to the decomposition relations between goals. However, the generated formulas may be incomplete, which needs to be corrected and verified by the requirements engineers. Our tool provides a formula correction interface to help requirement engineers complete the task of modifying and correcting formulas. The interface is shown in Fig. 11.

(4) Goal conflict detection. When the goals are represented by CTL formulas, the proposed goal conflicts detection method is used to identify conflicts between goals and show them in the graphical interface. In addition, the evidence about why the goals conflict is also given in the graphical interface. An example of conflict detection can be seen in Fig. 12. Fig. 12 shows the conflicting goals detected by the tool and their formal representation. Requirements engineers can modify the conflicting goals according to the results identified by the proposed method.

7. Experiments

We use a case study and controlled experiments to evaluate our proposed method. The objective of the evaluation is to assess whether our proposed method can efficiently and effectively support goal conflict detection.

To answer RQ1, we use a crossover service case from the industry to evaluate its effectiveness. The real-world industry case is the Fliggy crossover service, as mentioned above. Details of the experiments can be found in Section 7.1.

To answer RQ2, we use various crossover cases from the literature and industry partners and evaluate our proposed method in identifying goal conflicts. These cases contain different actors and goals and belong to multiple domains.

Case 1: Fliggy.[2] It combines a hotel service and an insurance service, thus reducing cancellation losses for users with a tentative itinerary.

Case 2: Cuntao.[3] It is an e-commerce platform that provides online shopping services to villagers in rural areas, which converges e-commerce services, logistics services, payment services, and insurance services.

Case 3: Smart agriculture (Zhang et al., 2020). The smart agriculture crossover service provides satellite remote sensing data to the agricultural domain to help farmers realize disaster prediction, precise planting, and other services.

Case 4: Eldercare crossover services (Guo et al., 2019b). It converges services from domains of healthcare, medical, and restaurant to satisfy the personalized care needs of elder people.

Case 5: 12 306.[4] It is China's largest train ticketing platform, which converges a train ticket service and a takeaway service to provide users with a good travel experience.

Case 6: Crossover healthcare service (Xi et al., 2019). It integrates services covering medical, eldercare, and healthcare fields to help the elders.

[2] www.fliggy.com.

[3] https://cun.taobao.com/.

[4] https://www.12306.cn.

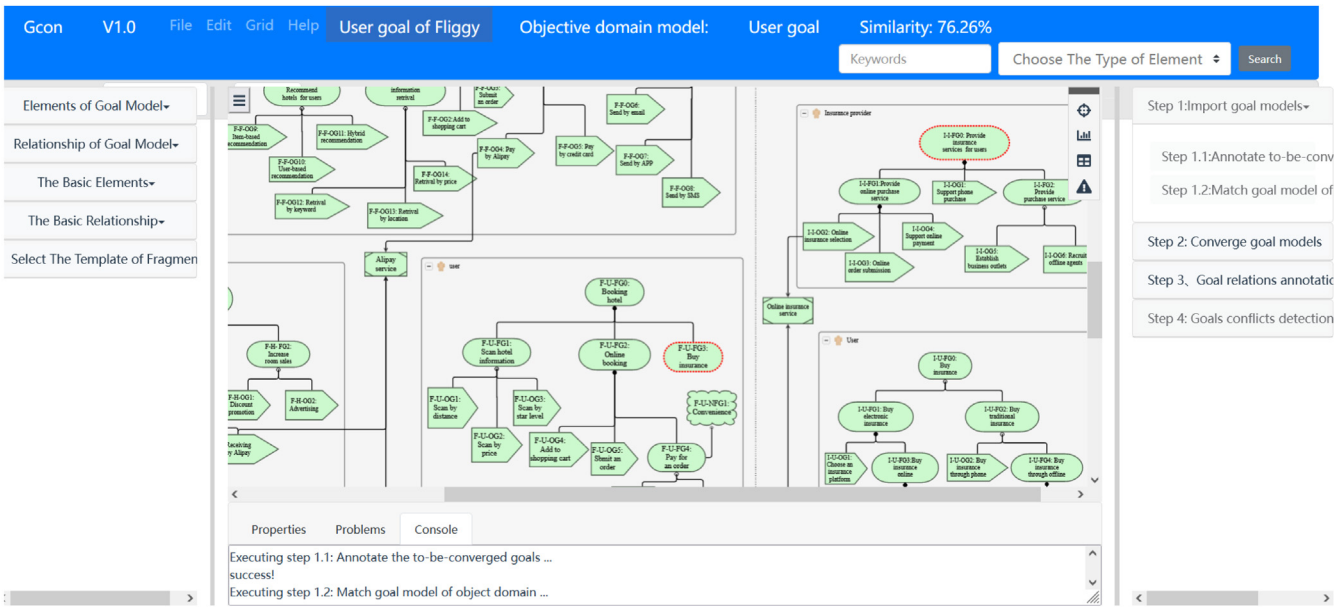

Fig. 9. Import goal model interface.

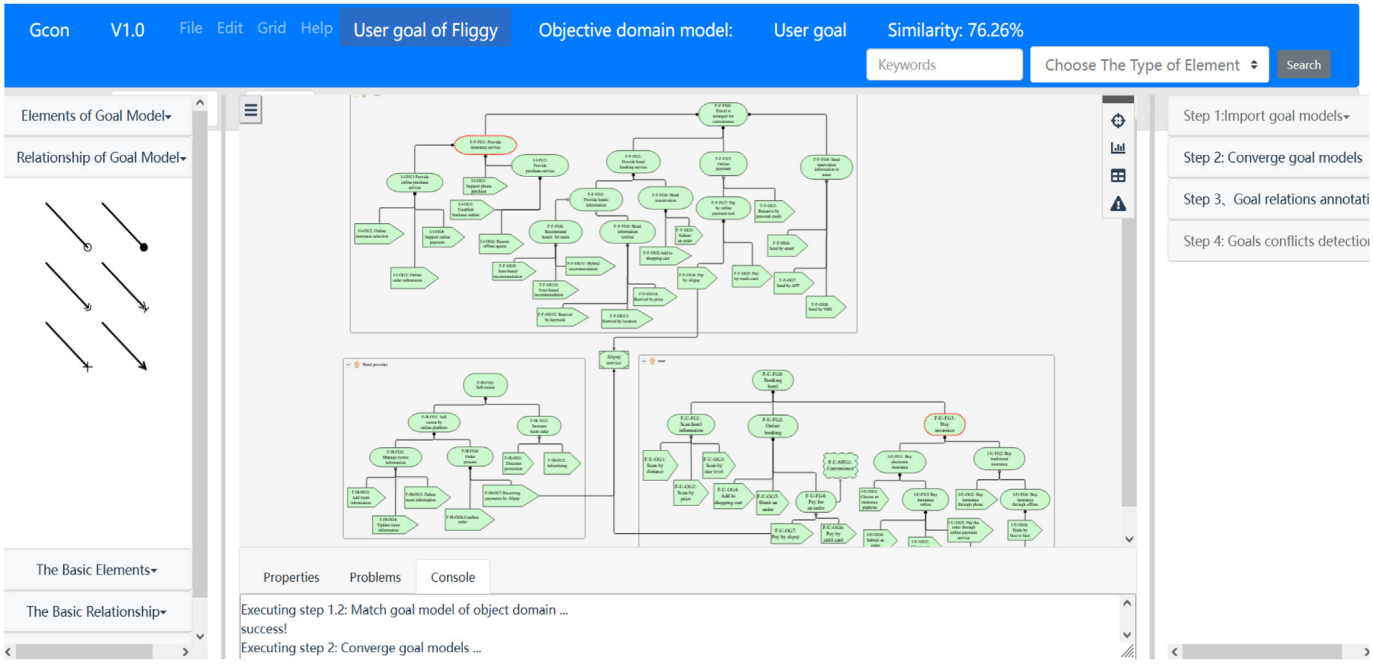

Fig. 10. Goal model convergence interface.

Case 7: Finance service for supply chains.[5] It converges manufacturing and finance services to provide convenient financing services for small and medium-sized enterprises.

To answer RQ3, we conducted a set of controlled experiments to evaluate our proposed method. We recruited subjects to participate in our experiments. The subjects used our tool to accomplish the whole experiment process. After that, they were requested to complete predefined questionnaires and participant in an interview related to the experiment. Details of the experiments can be found in Section 7.3.

5 https://www.hundsun.com.

7.1. Case study

We use the Fliggy case to evaluate the effectiveness of our proposed method in addressing the goal convergence of crossover services. We assume that two goal models of the travel domain and insurance domain already exist, as shown in Figs. 13 and 14. The goals of the actors involved in the two domains are presented and refined based on our previous work (Murukannaiah et al., 2015). Here, the task is to converge the two goal models of the two domains using our proposed goal convergence method, and thus the to-be-converged goal "Buy Insurance" in the subject

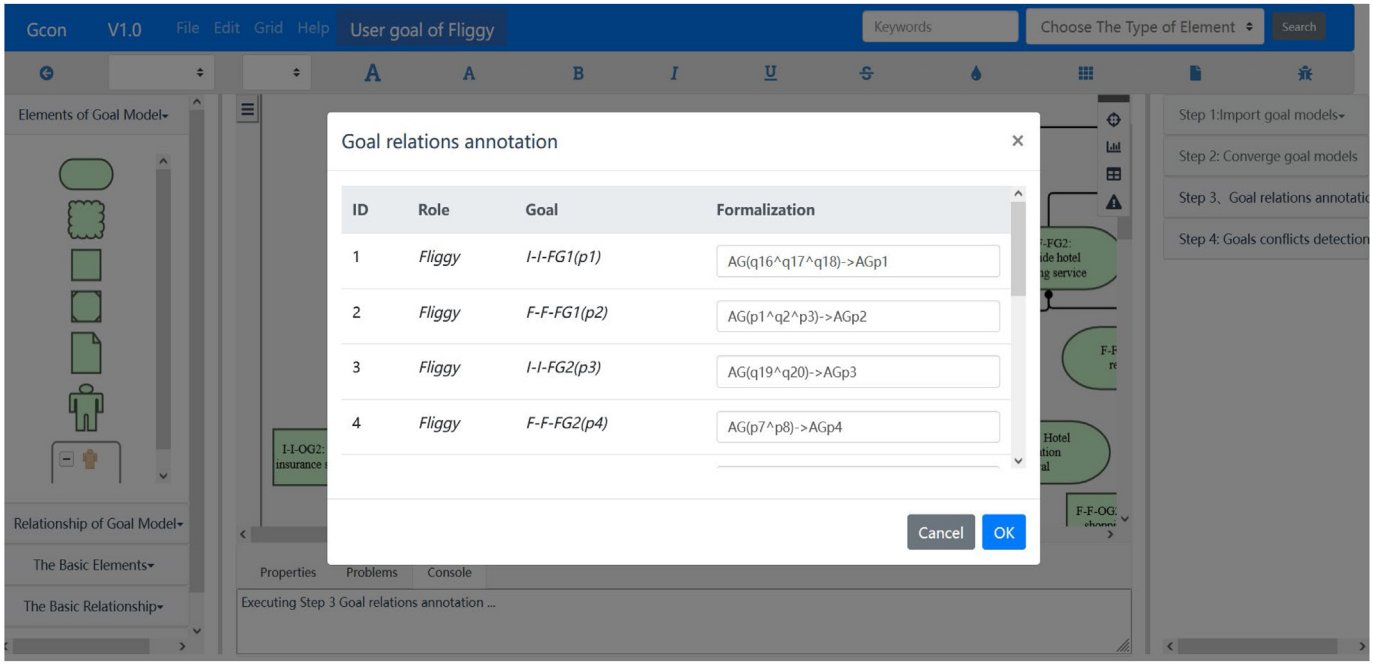

Fig. 11. Goal relations annotation interface.

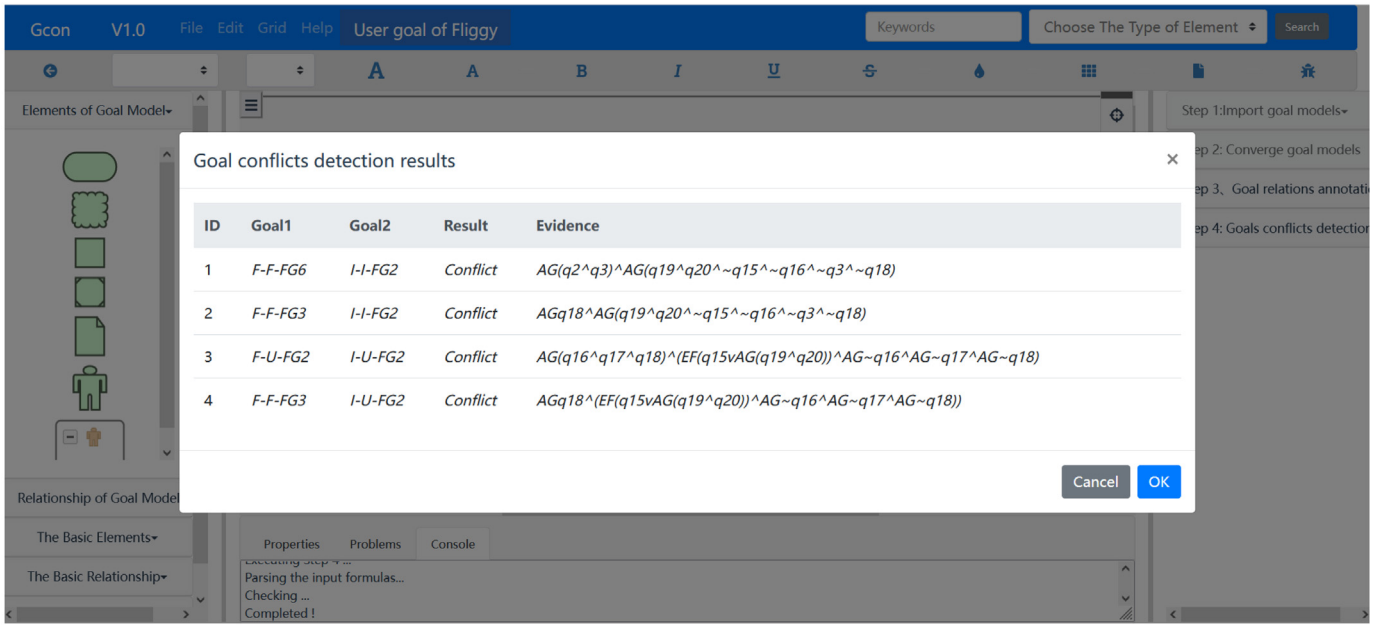

Fig. 12. Goal conflicts detection interface.

domain can be satisfied.

$$M_I = \begin{bmatrix} 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & -1 & -1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & -2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \quad (8)$$

First, according to the to-be-converged goal "Buy insurance" in the Fliggy domain, we find a similar goal "Buy insurance service" from the insurance domain based on the similarity calculation method illustrated in Section 4. Then the subgoals of the goal "Buy insurance service" should be converged into the Fliggy domain to make the to-be-converged goal "Buy insurance" satisfied, and the vertex of goal "Buy insurance service" is the convergence point.

To execute the convergence of two goal models, we use Algorithm 1 to extract decomposition paths from the convergence point "Buy insurance service", and we can get a set of decomposition paths $G_s = \{I\_U\_FG_1 \rightarrow I\_U\_OG_1 \rightarrow I\_U\_FG_3 \rightarrow I\_U\_OG_3 \rightarrow I\_U\_OG_4 \rightarrow I\_U\_OG_5 \rightarrow I\_U\_FG_2 \rightarrow I\_U\_OG_2 \rightarrow I\_U\_FG_4 \rightarrow I\_U\_FG_6\}$. Meanwhile, we extract the relations between goals in G_s and use a matrix M_I to store them.

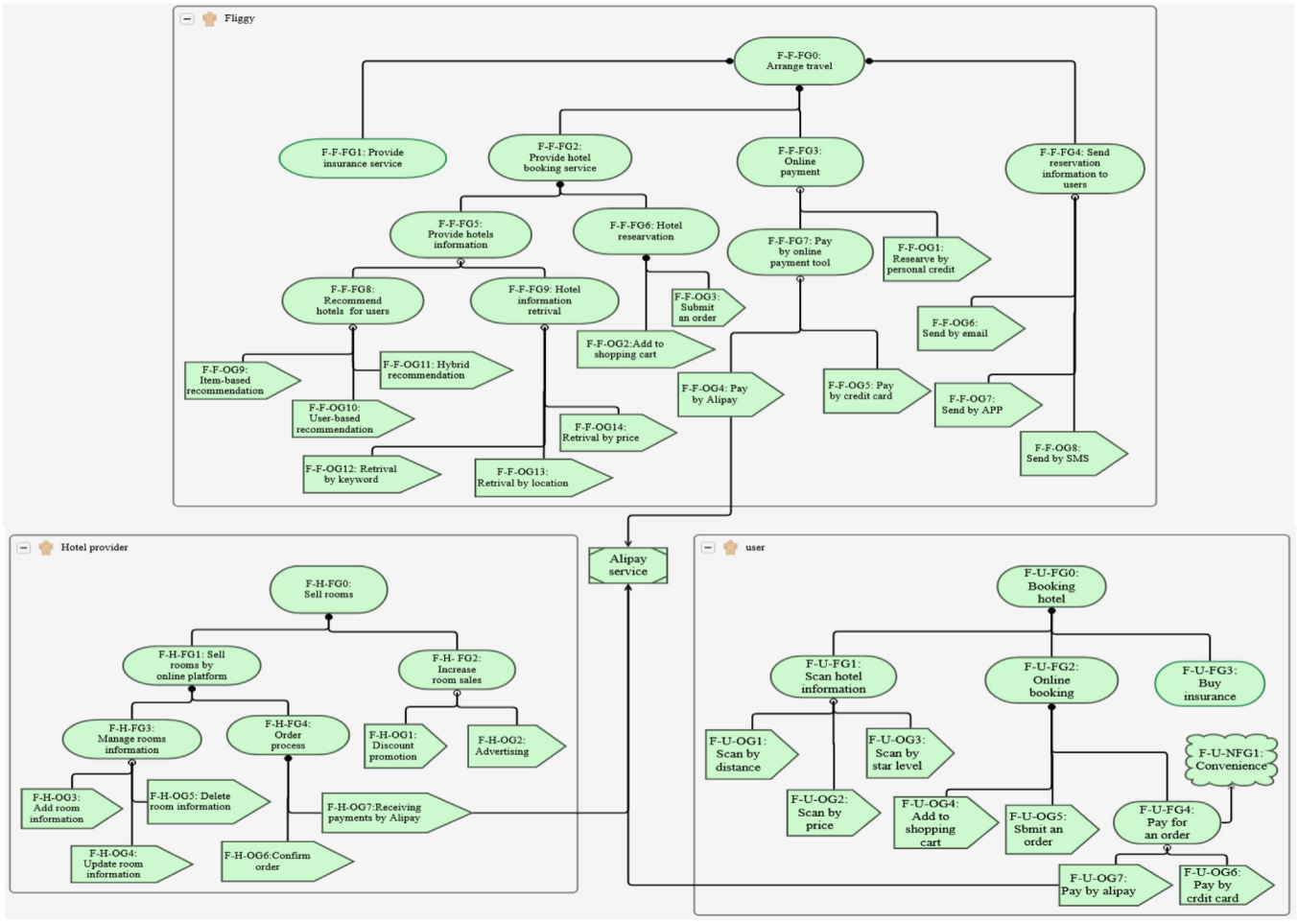

Fig. 13. Goal model in the Fliggy domain.

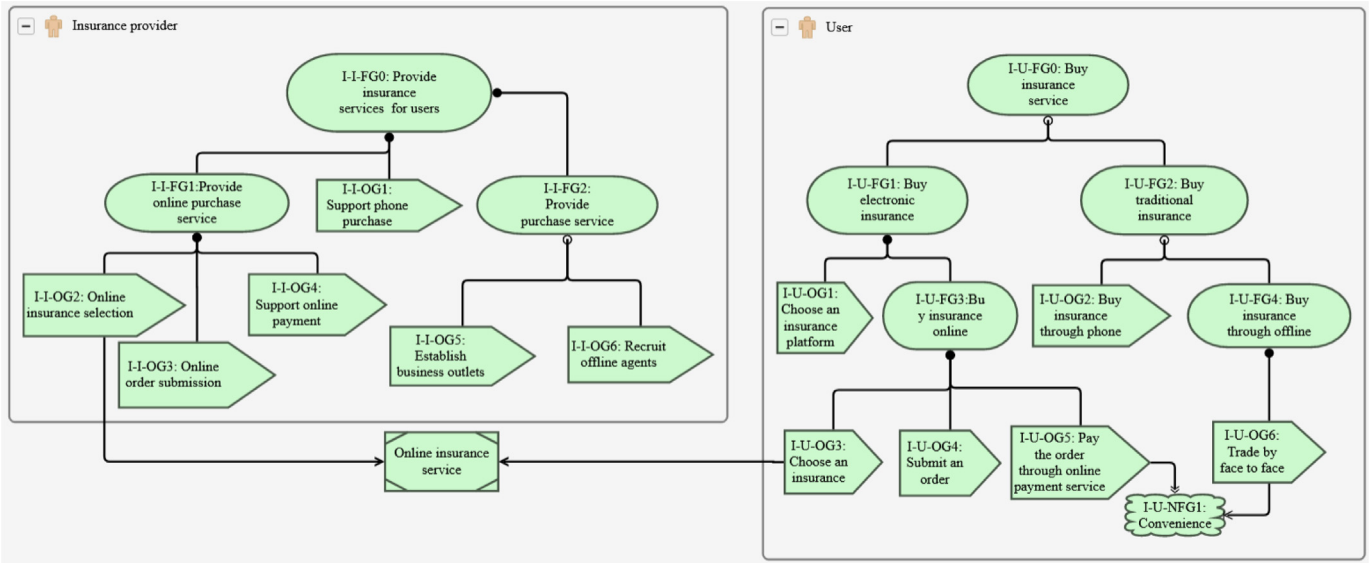

Fig. 14. Goal model in the insurance domain.

After we get the decomposition path G_s and the relation matrix M_I, we can converge the two goal models by our proposed convergence method and annotate the relations between converged goals, which are from the object domain according to the relation matrix M_I. The final converged goal model is shown in Fig. 15.

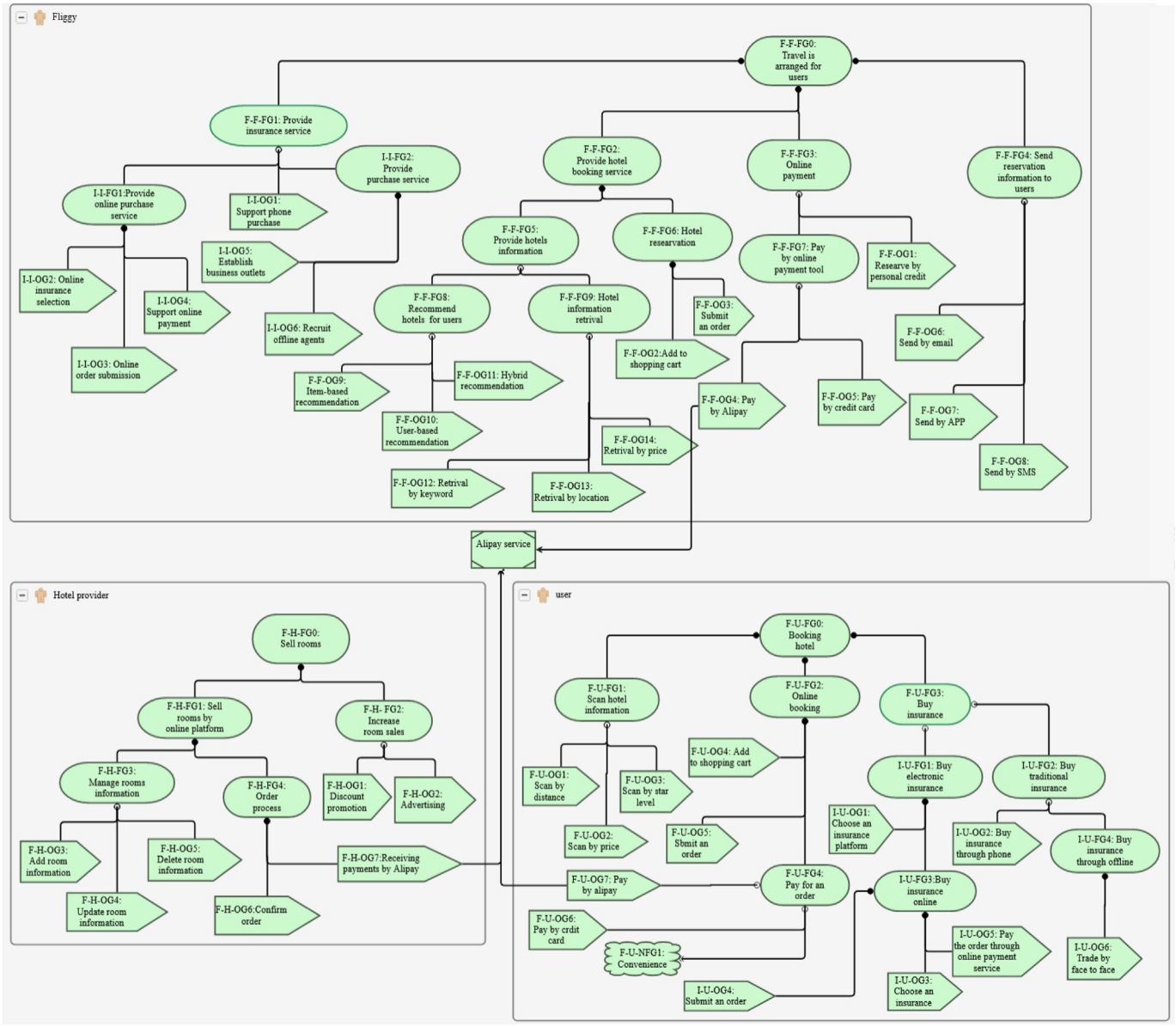

Fig. 15. The converged goal model of the user.

7.2. Performance of reasoning

After convergence, the detection tasks need to perform to identify conflicts between goals (RQ2). RQ2 mainly focuses on our proposed detection method's capability to identify conflicts that are not notable by molders. We report the model size of the converged goal model in terms of the number of actors (A), functional goals (FG), Non-functional goals (NFG), and operational goals (OG). The following columns relate to the results of applying our method to the detection of conflicts. In addition, we also report the required time in detecting the conflicts.

From the results shown in Table 4, we can see that our proposed method can detect conflicts between goals in the converged goal models. Our method has achieved good results in seven real crossover service cases. As the experimental evaluation shows, our proposed method can identify potential conflicts between goals for all the case studies of crossover services we applied. In the case of the small model size, e.g., the Meituan and 12 306, our method is efficient in finding a considerable amount of potential conflicts, which have not been recognized by the modelers who performed the task of goal convergence. For complex models like Fliggy and Cuntao, our method also performs well in identifying potential conflicts. In the case of Fliggy, our method detects four goal conflicts. Besides, our method detects three conflicts in the case of Cuntao. For other cases, like the medical crossover service and the finance crossover service, our method also detects four and three conflicts, respectively. As an answer to RQ2, our automated detection method can detect potential conflicts effectively.

7.3. Controlled experiments

To answer RQ3, we conducted a set of controlled experiments. First, we recruited subjects to participate in the experiments and gave scores to the predesigned questionnaires. To ensure the accuracy of the assessment, we selected subjects who are very familiar with crossover services and requirements engineering. After screening, we recruited 25 subjects to participate in our experiments, including six Ph.D. students, seven software engineers, ten postgraduate students, and two professors. All subjects major

Table 4
The analysis results of applying our method to seven crossover services.

Cases	Model size				Conflicts	
	A	FG	NFG	OG	Number	Time (s)
Fliggy	3	21	4	31	4	15.69
Cuntao	6	19	5	28	3	12.69
Meituan	3	10	5	19	2	8.79
12 306	5	7	3	13	5	9.68
Smart agriculture	3	11	2	16	3	10.26
Cross medical services	5	13	6	22	4	10.65
Supply chain finance	8	19	6	26	3	9.26

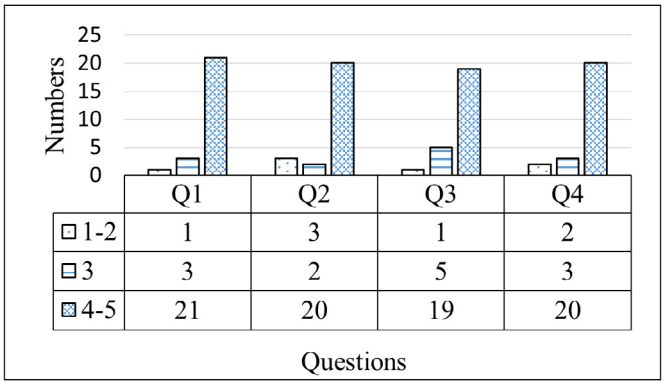

Fig. 16. Statistical results of the questionnaire.

in computer science. At the beginning of the experiment, we introduced some background of the case study and tool usage to these subjects. Then we introduced two representative crossover services. The first is the Meituan taxi service, introduced in the case study section. Another representative crossover service is Fliggy. Finally, these subjects were asked to use our developed tools to conduct requirements analysis on their selected cases and fill in the questionnaire within one day. To fully assess our proposed method, we set up a questionnaire with four questions to obtain meaningful and quantitative data. The questionnaire is listed below. The subjects were asked to fill in the questionnaires by giving a score to each question. The score range is from 1 to 5, where 1 represents Strongly Disagree, 3 represents Neutral, and 5 represents Strongly Agree.

1. It is simple to use the method to complete the tasks of goal convergence and conflict detection.
2. I could effectively complete the tasks of goal convergence and conflict detection using the method.
3. It is easy to understand the method.
4. Overall, I am satisfied with this method.

Question 1 mainly focuses on the usefulness of our proposed method, and Question 2 is used to evaluate the efficiency of the method. Question 3 is used to assess the understandability of the method. Question 4 assesses the overall satisfaction of subjects with the method. The results of the questionnaire are summarized in Fig. 16.

From the results shown in Fig. 16, we can see that more than 84% of the subjects gave positive comments on Question 1, which shows that the subjects think the proposed method has good usability, and most of them are satisfied with this method. It is worth noting that most of the subjects gave positive comments on Question 2, which shows that the proposed method can effectively help them complete requirements analysis tasks. Regarding Question 3, we can also see that the subjects gave positive evaluation scores, and about 80% of the subjects gave scores of 4 or

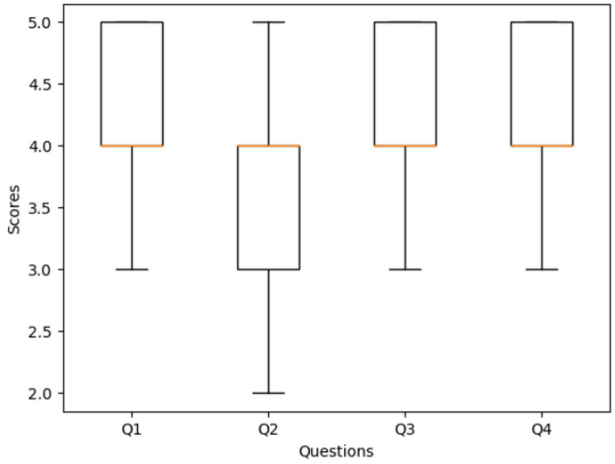

Fig. 17. The boxplot of scores of the questions.

5. We may conclude that our proposed method is very easy for users to understand. Note that 19 subjects gave 4 or 5 scores to Question 4, which shows that the subjects are generally satisfied with our method. The same conclusion also can be seen in Fig. 17. We can see from Fig. 17 that the median number of scores for the four questions is about 4, which indicates that participants gave a positive evaluation of the ease of use and efficiency of our method. Overall, we can conclude that our method has good feasibility from the results shown in Figs. 16 and 17.

After the subjects complete the questionnaires, we invite subjects to take part in additional interviews. We do not set predefined questions in advance, and the subjects are free to express their opinions. Note that only 20 subjects accept the interviews. Here we summarize the main concerns and suggestions raised by the subjects when answering the four questions of the interview:

- The relations between goals are difficult to define. It is a time-consuming task that should be supported by an automated tool. In addition, the relations defined by the molders may be incomplete.
- Adopting temporal relations between goals is helpful since this would contribute to identifying more conflicts. However, translating the relations between goals into CTL formulas is not easy, and it requires a deep understanding and rich experience of temporal logic. We still need to develop a systematic translation method via NLP or deep learning techniques. For example, we can convert the relations described in natural language into a standard CTL formula.
- Since a crossover service spans multiple domains, it requires modelers to be familiar with the subject and object domains, making it difficult for modelers to understand the domain knowledge.

These limitations suggest the direction of future work, particularly in the aspects of automation analysis and relations extraction.

7.4. Threats to validity

We discuss the threats to validity related to the evaluation of our proposed method. As suggested in many empirical software engineering studies (Runeson and Host, 2009; Feldt and Magazinius, 2010; Shull et al., 2008), we discuss the threats from the following four perspectives.

Conclusion validity: To ensure the reliability of the conclusions, we used multiple cases of different domains and model sizes to

validate our proposed goal conflicts detection method. In addition, we cross-validate our conclusions through questionnaires and interviews. Although our approaches can mitigate threats to conclusions, further validation in more realistic cases should be conducted in the future.

Construct validity: To mitigate the threat of construct validity, we draw our findings from various research methods: case studies, controlled experiments, and interviews. The case study mainly illustrates the approach and demonstrates its feasibility. During the interviews, participants are free to express any point they want to tell, which is not bound by predefined questions or answers. We record the key contents and confirm with participants at the end of the interview to avoid inaccuracy. We collect data from participants of multi-domains, thus avoiding a single bias in the study.

External validity: Our work is mainly oriented to the crossover service domain, and our findings are objective statements of the existing problems. To mitigate the threat of the small sample size and achieve reasonable generalizability, we increased the number of samples used for evaluation. These samples come from different industries and have different model sizes. The detection of goal conflicts was further assessed on seven crossover services from the industries or literature. Although these case studies offer good coverage of crossover services domains, generalization to other domains and larger models remains to be investigated. We still need to adjust and probably extend our modeling and reasoning framework to fit a range of application domains where the nature of requirements is different.

Internal validity: A major internal threat is that the modelers must specify relations between goals by CTL formulas, which is, to a certain degree, subjective. In our work, it is difficult to isolate issues related to goal modeling from those related to the novel mechanisms introduced in this paper. For example, using CTL formulas to express relations of goals should be done manually by requirements analysts with their experience and the understandability of the goal model. To mitigate this issue, we select participants familiar with crossover service and goal modeling. As part of the experiment, we explain the purpose of our study to the participants as a preliminary step to ask them to specify the relations between goals. This may affect the specification of the models and analysis input.

8. Conclusion and future work

This paper proposes two methods for goal convergence and its associated conflict detection techniques. We evaluate our proposed method from two perspectives. The first is about the effectiveness of our proposed method in dealing with the goal convergence of multiple domains. We use a real crossover service case from the industry to evaluate the goal convergence method and prove helpful. The second is the ability to detect conflicts between goals. Several crossover service cases were used to evaluate our proposed detection method, and the results demonstrate that our method can identify conflicts between goals in the converged goal model.

In the future, we plan to address problems in goal-based requirements engineering of crossover services. (1) Automated support of modeling and formula extraction. In this work, some procedures must be done manually by requirements analysts. For example, we extract CTL formulas from the converged goal models manually. Moreover, requirements analysts must manually specify the logical relations between goals, which is error-prone and time-consuming for large-scale systems. (2) Exploration for more detection methods. Besides CTL, there are many other logical mechanisms like model counting and SAT-based methods. We plan to explore other technologies in dealing with goal conflicts detection on a converged goal model.

CRediT authorship contribution statement

Zhengli Liu: Conceptualization, Methodology, Writing – original draft. **Bing Li:** Funding acquisition, Supervision, Project administration. **Jian Wang:** Writing – review & editing. **Xiangfei Lu:** Software. **Yu Qiao:** Validation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (Nos. 62032016 and 61832014) and the Key Research and Development Program of Hubei Province, China (No. 2021BAA031).

References

Ali, R., Dalpiaz, F., Giorgini, P., 2010. A goal-based framework for contextual requirements modeling and analysis. Requir. Eng. 15 (4), 439–458.
Ali, R., Dalpiaz, F., Giorgini, P., 2013. Reasoning with contextual requirements: Detecting inconsistency and conflicts. 55, (1), pp. 35–57,
Asadi, M., Gröner, G., Mohabbati, B., Gašević, D., 2016. Goal-oriented modeling and verification of feature-oriented product lines. Softw. Syst. Model. 15 (1), 257–279.
Baslyman, M., Amyot, D., 2019. Goal model integration: Advanced relationships and rationales documentation. In: 2019 International Conference on System Analysis and Modeling. pp. 183–199.
C. Edmund, M., Jr, O., Daniel, K., Helmut, V., 2000. Model Checking. MIT Press, Cambridge.
Chatzikonstantinou, G., Kontogiannis, K., 2018. Efficient parallel reasoning on fuzzy goal models for run time requirements verification. Softw. Syst. Model. 17 (4), 1339–1364.
Chen, X., Cao, H., Ye, L., Liang, Z., 2020. Value innovation with crossover services. In: 2020 IEEE World Congress on Services (SERVICES) Value. pp. 237–244.
Clarke, E., Henzinger, T., Veith, H., Bloem, R., 2018. Handbook of model checking.
Dalpiaz, F., Franch, X., Horkoff, J., 2016. Istar 2.0 language guide. arXiv, http://arxiv.org/abs/1605.07767.
Degiovanni, R., Castro, P., Arroyo, M., Ruiz, M., Aguirre, N., Frias, M., 2018a. Goal-conflict likelihood assessment based on model counting. In: Proceedings - 2018 ACM/IEEE 40th International Conference on Software Engineering. pp. 1125–1135.
Degiovanni, R., Regis, G., Molina, F., Aguirre, N., 2018b. A genetic algorithm for goal-conflict identification. In: ASE 2018 - Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering. pp. 520–531.
Degiovanni, R., Ricci, N., Alrajehy, D., Castro, P., Aguirre, N., 2016. Goal-conflict detection based on temporal satisfiability checking. In: ASE 2016 - Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering. pp. 507–518.
Drechsler, R., Sieling, D., 2001. Binary decision diagrams in theory and practice. Int. J. Softw. Tools Technol. Transf. 3 (2), 112–136.
Emerson, E., Clarke, E., 1982. Using branching time temporal logic to synthesize synchronization skeletons. Sci. Comput. Program. 2 (3), 241–266.
Feldt, R., Magazinius, A., 2010. Validity threats in empirical software engineering research - An initial survey. In: Proceedings of the 22nd International Conference on Software Engineering and Knowledge Engineering. pp. 374–379.
Giorgini, P., Mylopoulos, J., Nicchiarelli, E., Sebastiani, R., 2002. Reasoning with goal models. In: 21st International Conference on Conceptual Modeling. pp. 167–181.
Giorgini, P., Mylopoulos, J., Sebastiani, R., 2005. Goal-oriented requirements analysis and reasoning in the Tropos methodology. 18, (2) pp. 159–171,
Guo, S., Xu, C., Chen, S., Xue, X., Feng, Z., 2019a. Crossover fusion approach for health services based on microservice architecture. In: 2019 IEEE World Congress on Services, SERVICES 2019. pp. 237–241.

Guo, S., Xu, C., Chen, S., Xue, X., Feng, Z., Chen, S., 2019b. Crossover service fusion approach based on microservice architecture. In: Proceedings - 2019 IEEE International Conference on Web Services, ICWS 2019. pp. 237–241.
Letier, E., Van Lamsweerde, A., 2004. Reasoning about partial goal satisfaction for requirements and design engineering. In: Proceedings of the ACM SIGSOFT Symposium on the Foundations of Software Engineering. pp. 53–62.
Li, M., Tu, Z., Xu, X., Xu, H., Wang, Z., 2020. A collaborative modeling approach for crossover services. In: 2020 IEEE World Congress on Services. SERVICES, pp. 219–224.
Liu, Z., Li, B., Wang, J., Qiao, Y., 2020. A value-driven modeling approach for crossover services. Int. J. Web Serv. Res. 17 (3), 20–38.
Murukannaiah, P., Kalia, A., Telang, P., Singh, M., 2015. Resolving goal conflicts via argumentation-based analysis of competing hypotheses. In: 2015 IEEE 23rd International Requirements Engineering Conference, RE 2015 - Proceedings. pp. 156–165.
Nguyen, C., Sebastiani, R., Giorgini, P., Mylopoulos, J., 2018. Multi-objective reasoning with constrained goal models. Requir. Eng. 23 (2), 189–225.
Peng, Y., Li, B., Wang, J., Liu, Z., 2020. An approach of crossover service goal convergence and conflicts resolution. In: 2020 IEEE World Congress on Services. SERVICES, pp. 225–230.
Pinna Puissant, J., Van Der Straeten, R., Mens, T., 2013. Resolving model inconsistencies using automated regression planning. Softw. Syst. Model. 14 (1), 461–481.
Qian, W., Peng, X., Wang, H., Mylopoulos, J., Zheng, J., Zhao, W., 2018. MobiGoal: Flexible achievement of personal goals for mobile users. IEEE Trans. Serv. Comput. 11 (2), 384–398.
Reimers, N., Gurevych, I., 2019. Sentence-BERT : Sentence embeddings using siamese BERT-networks. In: Proceedings Ofthe 2019 Conference on Empirical Methods in Natural Language Processing. pp. 3982–3992.
Runeson, P., Host, M., 2009. Guidelines for conducting and reporting case study research in software engineering. Empir. Softw. Eng. 14, 131–164.
Shan, Y., Qiao, Y., Li, B., Wang, J., 2020. A process convergence approach for crossover services based on message flow partition and merging. In: 2020 IEEE International Conference on Services Computing. pp. 178–185.
Shull, F., Singer, J., Sjoberg, D., 2008. Guide to Advanced Empirical Software Engineering, Vol. 4. Springer, London, London, UK, (1).
Tarjan, R., 1972. Depth-first search and linear graph algorithms. SIAM J. Comput. 1 (2), 146–160.
Ulfat-Bunyadi, N., Gol Mohammadi, N., Wirtz, R., Heisel, M., 2019. Systematic Refinement of Softgoals Using a Combination of KAOS Goal Models and Problem Diagrams, Vol. 1077. Springer International Publishing.
Wang, F., 2004. Efficient verification of timed automata with BDD-like data structures. Int. J. Softw. Tools Technol. Transf. 6 (1), 77–97.
Xi, M., et al., 2019. A scenario-based requirement model for crossover healthcare service. In: 2019 IEEE World Congress on Services, SERVICES 2019. pp. 252–259.
Xue, X., Gao, J., Wu, S., Wang, S., Feng, Z., 2021. Value based analysis framework of crossover service: A case study of new retailer in China. IEEE Trans. Serv. Comput. 15 (1), 83–96.
Yin, J., et al., 2018. Crossover service: Deep convergence for pattern, ecosystem, environment, quality and value. In: Proc. - Int. Conf. Distrib. Comput. Syst. 2018-July. pp. 1250–1257.
Zhang, X., Ma, S., Su, C., Shang, Y., Wang, T., Yin, J., 2020. Coastal Oyster Aquaculture Area extraction and nutrient loading estimation using a GF-2 satellite image. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 13, 4934–4946.

Zhengli Liu is currently a Lecture in the School of Computer Science and Artificial Intelligence, Wuhan Textile University, Wuhan, China. His current research interests include services computing and software engineering.

Bing Li is currently a Professor in the School of Computer Science, Wuhan University, Wuhan China. His current research interests include services computing and software engineering.

Jian Wang is currently an associate professor in the School of Computer Science, Wuhan University, Wuhan, China. His current research interests include software engineering and services computing.

Xiangfei Lu is currently working toward the M.S. degree in software engineering with Wuhan University, Wuhan, China. Her current research interests include services computing and software engineering.

Yu Qiao is currently a Ph.D. student in the School of Computer Science, Wuhan University, Wuhan, China. Her current research interests include services computing and software engineering.