Correcting Interaction Mismatches for Business Processes

Shuai Gong\textsuperscript{1,2} Jinhua Xiong\textsuperscript{1} Zhiyong Liu\textsuperscript{1} Cheng Zhang\textsuperscript{1,2}

1. Institute of Computing Technology, Chinese Academy of Sciences, 2. Graduate University of Chinese Academy of Science
E-mail: gongshuai@software.ict.ac.cn, \{xjh, zylwu\}@ict.ac.cn, zhangcheng2@software.ict.ac.cn

Abstract

Discovering and correcting interaction mismatches in inter-organizational business processes are important for service choreography. Automatic discovery mismatches and providing correcting plans will alleviate burdens on business modelers, however, three major challenges are: 1) by what cost model to evaluate multiple correcting plans; 2) how to avoid generating new mismatches when correcting processes; 3) how to effectively reduce the search space size. In this paper, we first extend our previous approach to discover interaction mismatches, and present a cost model for evaluating correcting plans; secondly, we propose an independent modifying region-based method to obtain multiple correcting plans which can avoid generating new mismatches, and reduce the search space for finding out correcting plans. A running example is given to illustrate the validity.

1. Introduction

Inter-organizational business processes are used to model complex interaction behavior between processes of different partners. When designing inter-organizational business processes, it is critical to ensure that these processes are able to interact properly, i.e. interactions are compatible. Although, there are many works which aim at verification of compatibility between processes (See \cite{8}[9][10][11]), few work has been conducted to identify which mismatched interaction pairs cause business processes mismatched\cite{4}, or to correct mismatched process \cite{1}.

When business processes are discovered incompatible, most of existing approaches tried to generate adaptors to overcome mismatch problems (See \cite{2}[3][7][8]). To the best of our knowledge, only the work in \cite{1} tries to support automatically fixing of choreographies to some extent. Lohmann presents an edit distance (i.e., the actions needed to fix the service) to express the cost of correcting mismatched interactions \cite{1}. But correcting mismatched interactions might not be characterized by the detailed edit distance. More business knowledge and/or strategies should be considered. One effective approach is to provide multiple correcting plans to the business modelers, and let them to make the final decision and correct the mismatched interactions. The cost of correcting plans indicates the effort of correcting processes, so it makes sense to recommend multiple correcting plans with minimal cost to business modelers. In order to provide multiple correcting plans with minimal cost, three major issues need to be addressed:

1) How to measure the cost of correcting plans. The cost model will not step into the details of basic edit operations, but will reflect the cost of high-level operations. Meanwhile, it must be easy to calculate.

2) How to avoid generating new mismatches when correcting mismatched business processes.

As we can see from Fig.1, process $P$ only contains three activities and two transitions, so does process $Q$. Initially there are only two mismatched interaction pairs: ($i_1$, $i_3$), ($i_2$, $i_3$).When correcting mismatched interaction pair ($i_1$, $i_3$), we remove the transition from activity $d$ to activity $e$ and add the transition from activity $f$ to activity $d$. Although the correcting plan will make the interaction pair ($i_1$, $i_3$) compatible, but it will make interaction pair ($i_1$, $i_2$) mismatched.

![Figure 1. Correcting a mismatched interaction pair generates a new mismatched interaction pair.](image)

3) When the size of interactions of business process is large, the search space size for finding out multiple correcting plans is very large, which is approximately exponential \cite{1}. Then, how to search effectively is a critical issue.

In order to address the above-mentioned issues, this paper first presents a cost model for correcting plans based on the size of modified structure relations.

And to avoid generating new mismatches illustrated in Fig.1, an IMR-based correcting approach for interaction mismatches is proposed. IMR is an abbreviation of Independent Modifying Region. Correcting all the mismatched interaction pairs in an IMR will not generate new mismatched interaction pairs.
In order to reduce the size of search space, two effective ways are adopted. Firstly, by identifying all IMRs, correcting plans could be based on IMRs, this leads to two effects: reducing search space size by isolating each IMR; meanwhile some mismatched interaction pairs might be removed from the search space. Secondly, by recognizing specific patterns within each IMR, pattern-based correcting plans could be pre-built for each pattern; this also helps to reduce the size of search space.

In a summary, this paper presents an approach to discover interaction mismatches, and to find out multiple minimal cost correcting plans for interaction mismatches, which are evaluated by the size of modified structure relations. The approach alleviates the complexity of correcting mismatched business processes, and the search space size for multiple correcting plans is less than the classical breadth-first algorithm.

The remainder of this paper is organized as follows. Section 2 reviews the related works. Section 3 extends the mismatch discovery approach based on our precious work [4]. Section 4 discusses the correcting plans for mismatched interactions, including the cost model for correcting plans, identifying independent modifying region and independent modifying region based correcting plans recommendation. Section 5 evaluates the result with a running example. Section 6 concludes the paper and outlines the future work.

2. Related Work

Many techniques have been proposed to verify compatibility of business processes (See [4][8][9][10][11]). Bordeaux reviews related work about business process compatibility verifications [5]. Two typical behavioral mismatches which are message order error and missing/extra message mismatch (See Fig.3) are summarized in [2]. The existing approaches (See [4][8][9][10][11]) are able to discover both message order errors and missing/extra message mismatches. The mismatch discovery approach proposed in [4] is not only able to decide whether business processes mismatched but also able to find out which interactions cause the processes mismatched. Therefore, the approach presented in [4] will be used to discover mismatches between processes in the paper.

When business processes are mismatched, there are two typical ways to overcome the problem: automatically or semi-automatically generating adaptors (See [2][3][7][8]) and modifying business processes(See [1]). Mismatch patterns in both operation and protocol level are summarized in [2], adaptors for these mismatch patterns are also generated automatically. Automatic ways to generate adaptors are also presented in [7][8]. A semi-automatic way to generate adaptors for deadlock interactions is presented in [3], but it fails to deal with several independent deadlocks. To the best of our knowledge, these approaches mainly adapt mismatches in the operation level (or interface level), they cannot adapt behavioral mismatch well, such as deadlock interactions. Lehmann tries to modify business processes to remove mismatches between them [1], but only provides one correcting plan. So business modelers have no other choices in correcting business processes. In a summary, the existing approaches of overcoming mismatches between processes either fail to adapt behavioral mismatches well, or ignore the effort of business domain knowledge in correcting mismatched business processes.

Li, Rechert, and WombacherOn present a way to measure the similarity of businesses processes based on high-level operations instead of basic edit operations [12], which inspired us to propose a new cost model for correcting plans based on the modified structure relations. Weidlich, Weske and Mendling present a way to identify the region needed to be modified in the target process model according to the changes in the source process model [13]. This inspired us to identify independent modifying region in the mismatched processes before finding out correcting plans.

3. Discovering Interaction Mismatches

This section extends our previous work [4], presents how to discover interaction mismatches based on interaction relations embodied by engaged activities’ structure relations.

![Activity Structure Relation](image)

A business process can be formalized as a triple (A, T, Re), while A indicates the activity set of the business process, T indicates the transition set of the business process and Re indicates the structure relation set between activities of the business process. The structure relations between activities in the business process can be classified into six types, which are extended from five types in the previous work [4]: Sequence Previous (S_Pre), Sequence Post (S_Post), Parallel (PL), Complete Exclusive (Com_E), Partly Exclusive Previous(Part_E_Pre) and Partly Exclusive Post(Part_E_Post).
A message transmitting between processes is an interaction, which can be indicated as a tuple \( I = \langle a, b \rangle \), where \( a, b \) indicate activities from different business processes. The structure relation between any two interactions consists of different structure relations of participating activities in each business process.

For example in Figure 3(1), there is a pair of interactions \( (i_1 = \langle b,c \rangle, i_2 = \langle a, d \rangle) \) between two processes. The activity engaged in \( i_1 \) in process \( P \) is \( b \), and the activity engaged in \( i_2 \) in process \( Q \) is \( c \). Therefore, the structure relation between interaction \( i_1 \) and \( i_2 \) in process \( Q \) is \( c \text{ S}_\text{Pre} d \). Meanwhile the activity engaged in \( i_1 \) in process \( Q \) is \( c \), and the activity engaged in \( i_2 \) in process \( Q \) is \( d \), the relation between \( c \) and \( d \) in process \( Q \) is \( c \text{ S}_\text{Post} d \). Therefore, the structure relation between interaction \( i_1 \) and \( i_2 \) in process \( Q \) is \( i_1 \text{ S}_\text{Post} i_2 \). Put it together, the interaction relation can be indicated as: \( IR_{R_0} (i_1,i_2) <= i_1\text{ S}_\text{Post} i_2, i_1\text{ S}_\text{Pre} i_2 >= i_2 \text{ S}_\text{Post} i_2 \).

\[ \begin{array}{cccccc}
S_\text{Pre} & S_\text{Post} & \text{PL} & \text{Com}_E & \text{Part}_E & \text{Part}_E \\
\text{C} & \text{MIS} & \text{C} & \text{MIS} & \text{MIS} & \text{MIS} \\
\text{MIS} & \text{C} & \text{C} & \text{MIS} & \text{MIS} & \text{MIS} \\
\text{C} & \text{C} & \text{C} & \text{MIS} & \text{MIS} & \text{MIS} \\
\text{MIS} & \text{MIS} & \text{MIS} & \text{MIS} & \text{MIS} & \text{MIS} \\
\text{MIS} & \text{MIS} & \text{MIS} & \text{MIS} & \text{MIS} & \text{MIS} \\
\end{array} \]

**Figure 3. Two Typical Behavioral Mismatches**

Inspired by the works in [2][3], the behavioral mismatches between business processes can be reduced to two types: Message Order Error and Missing/Extra Message, as illustrated in Figure 3. If any two interactions cause the above mismatches, they are called as Mismatched Interaction Pair (MIP). A mismatched interaction pair \( (i_1, i_2) \) can be indicated as \( i_1\text{Mi}_2 \). Whether a pair of interactions is mismatched or not, can be concluded according to Tab. 1. Therefore, Complete Interaction Mismatch Set (CIMS), which consists of all mismatched interaction pairs between two processes, can be obtained by comparing structure relations of any two interactions.

According to the Tab. 1, and considering interaction structure relations, there are 7 cases in which interaction pairs are compatible, and there are 29 cases in which interaction pairs are mismatched. Compared with [4], Tab. 1 identifies more fine-grain mismatch cases. For example, for the case of \( IR_{R_0} (i_1,i_2) = \langle \text{Com}_E, \text{Com}_E \rangle \) marked by *, the interactions could be compatible if satisfying certain conditional constraints, therefore, this case could also be provided as a candidate correcting plan to the business modelers.

**Table 1.** The first row indicates the relation between \( i_1 \) and \( i_2 \) in process \( P \), and the first column indicates the relation of \( i_1 \) and \( i_2 \) in process \( Q \). C indicates \( i_1 \) and \( i_2 \) are compatible, and MIS indicates \( i_1 \) and \( i_2 \) are mismatched. * indicates these interactions probably will not cause business processes fail to interact properly under certain conditional constraint. In this paper, if no special explanation, these interactions are considered mismatched.

**4. Correcting plans for Interaction Mismatches**

This section discusses how to find out multiple minimal cost correcting plans when business processes are mismatched. The section begins with a new cost model for evaluating multiple correcting plans based on modified structure relations, and then discusses how to identify independent modifying region, at last discusses how to find out multiple correcting plans for each independent modifying region, which will be recommended to the business modelers.

**4.1. New Cost Model for Correcting Plan**

This section discusses the new cost model for correcting plans, which evaluate the cost of correcting plan based on modified structure relations.

Correcting mismatched interactions is not trivial as indicated by the basic edit operation distance, more business knowledge and/or strategies should be considered. Therefore, the new cost model does not step into the level of basic edit operations, but reflect, to some extent, the cost of business-level operations, i.e. evaluate the cost of correcting plans by the size of modified structure relations.

According to Tab. 1, if a mismatched interaction pair needs to be corrected, there will be 7 candidate correcting plans, because there are 7 cases that a pair of interaction is compatible. Furthermore, there may exist more than one correcting plans with minimal cost for mismatched pattern (See Sect.4.3). This is a side-effect of the new cost model that it could provide diversity for minimal cost correcting plans.

Correcting plans indicate how to correct business processes [1]. The basic edit operations for business processes include adding, deleting, moving and updating. Instead of using these basic edit operations to describe
and evaluate the cost of correcting plans, this paper uses modified structure relations to describe correcting plans and the size of modified structure relations to evaluate the cost of them. In this paper, correcting plans consist of a set of modified structure relations of interactions. The correcting plans can be expressed as follows:

\[
IR_{P,Q}(i_1, i_2) = < r_1, r_2 > \lor < r_3, r_4 >
\]

\[
IR_{P,Q}(i_3, i_4) = < r_5, r_6 > \lor < r_7, r_8 > \ldots
\]

In the above while \(i_1, i_2, i_3, i_4\) indicate the interactions, which relations will be modified, and \(r_1, r_2, r_3, r_4, r_5, r_6, r_7, r_8\) indicate the structure relations between interaction pairs. \(IR_{P,Q}(i_1, i_2) = < r_1, r_2 > \lor < r_3, r_4 >\) means that the relations between interaction pair \((i_1, i_2)\) will be modified from \(<r_1, r_2>\) to \(<r_3, r_4>\). For example, correcting mismatched interaction pair \((i_3, i_1)\) in Fig.1 can be described as: \(IR_{P,Q}(i_3, i_1) = < S_\text{Pre}, S_\text{Post} > \lor < S_\text{Pre}, S_\text{Pre} >\) and the cost of the correcting plan here is 1.

As illustrated above, the new cost model here is straightforward and independent of any process modeling techniques. Really effort for a business analyst to modify business processes is to re-design or re-implement the activities and relations among them, so the size of modified structure relations could better characterize the cost of correcting plans at high-level. And because it might not distinguish between the types of relation, the order inside relations, the cost model also supports the diversity of correcting plans.

4.2. Identifying Independent Modifying Region

This section discusses the definition of independent modifying region and the identifying algorithm for it. After discovering all the mismatched interaction pairs between processes, the default method of finding out correcting plans is classical breadth-first search algorithm. However, the search space for this algorithm is large and the problems shown in Fig.1 will happen. The motivation of identifying independent modifying region is to solve the last two problems mentioned in Sect.1.

For example, in Fig.4 there are 11 mismatched interaction pairs and 2 independent modifying regions. After identifying independent regions, the mismatched interaction pairs \((i_1, i_2), (i_1, i_3), (i_4, i_2), (i_4, i_3), (i_5, i_2), (i_5, i_3), (i_6, i_2), (i_6, i_3)\) will be removed from the search space. And the problem shown in Fig.1 can also be avoided. That is because mismatched interaction pairs, one of which is corrected can probably generate new mismatched interaction pairs, are always in the same independent modifying region. For space limitation, the details are neglected.

4.2.1. Independent Modifying Region Definition

Definition 1: Independent modifying region (IMR) consists of mismatched interaction pairs and two well-structured [6] interacting sub processes, and IMR must satisfy the following two conditions. Independent modifying region can be formalized as: \(\text{IMR} = < P', Q', \text{MIPs}>\), while \(P'\) indicates the well-structured sub process of process \(P\), \(Q'\) indicates the well-structured sub process of process \(Q\) and \(\text{MIPs}\) indicates the mismatched interaction pairs in this region.

**Condition 1**: After correcting all the mismatched interaction pair in an IMR, there will be no interactions outside of this IMR which are mismatched with interactions in this IMR.

\[(\forall i)(\neg \exists i')(i \in \text{IMR} \land i' \notin \text{IMR} \land (iMi'))\]

**Condition 2**: Correcting all the mismatched interaction pairs in an IMR will not generate new mismatched interaction pairs.

\[\neg \exists i'(\exists \neg i'((ii') \land (i', i') \notin \text{CIMS}))\]

In order to identify independent modifying region, Theorem 1 and the concept of the SEED of an IMR and is proposed.

**Theorem 1**: For any mismatched interaction pair \((i_1, i_2)\), at least one interaction from the pair must be in an IMR.

\[
\begin{align*}
IR_{P_{\text{IMR}}}(i_1, i_2) &= < S_\text{Pre}, S_\text{Post} >, \\
IR_{P_{\text{IMR}}}(i_1, i_2) &= < S_\text{Post}, S_\text{Pre} >, \\
IR_{P_{\text{IMR}}}(i_1, i_2) &= < \text{Part}_E_\text{Pre}, \text{Part}_E_\text{Post} >, \\
IR_{P_{\text{IMR}}}(i_1, i_2) &= < \text{Part}_E_\text{Post}, \text{Part}_E_\text{Pre} >, \\
IR_{P_{\text{IMR}}}(i_1, i_2) &= < S_\text{Pre}, \text{Part}_E_\text{Post} >, \\
IR_{P_{\text{IMR}}}(i_1, i_2) &= < \text{Part}_E_\text{Post}, S_\text{Pre} >, \\
IR_{P_{\text{IMR}}}(i_1, i_2) &= < \text{Part}_E_\text{Pre}, S_\text{Post} >, \\
IR_{P_{\text{IMR}}}(i_1, i_2) &= < S_\text{Post}, \text{Part}_E_\text{Pre} > \\
\end{align*}
\]

**Figure 4. Independent Modifying Region**

![Figure 4. Independent Modifying Region](image-url)
\[ IR_{\text{eq}}(i_1, i_2) = \langle \text{Com}_E, \ast >, \quad \text{Group2} \]

**Definition 2**: SEED of an IMR is a mismatched interaction pair \((i_1, i_2)\) when interactions \(i_1\) and \(i_2\) must be contained in one IMR, which means the two interactions must be modified together, in order to avoid generating new mismatches.

In order to decide which mismatched interaction pairs are the SEED of IMRs, Theorem 2 are proposed.

**Theorem 2**: If structure relation between mismatched interaction pair \((i_1, i_2)\) is one of the interaction relations in Group1, or in Group2, the interaction pair \((i_1, i_2)\) is the SEED of an IMR.

Because Theorem 2 is crucial to decide which mismatched interaction pair is not the seed of any IMR, so there are three cases as follows.

1) Suppose that the structure relation between them is one of the interaction relations in Group1 and the interaction pair is not the seed of any IMR, it is proven next.

**Proof**: There is a mismatched interaction pair
\[i_1 < a, b >, i_2 < c, d >; a, c \in P.A \land b, d \in Q.A\]

1) Suppose that the structure relation between them is one of the interaction relations in Group1 and the interaction pair is not the seed of any IMR, so there are three cases as follows.

- \(i_1 \in IMR_1 \land i_2 \in IMR2\). When correcting IMR1 or IMR2, the mismatched interaction pair \((i_1, i_2)\) will not be corrected since \(i_1, i_2\) are not in a same IMR. So after IMR1 or IMR2 are corrected, interaction pair \((i_1, i_2)\) is still mismatched, which conflicts with the **Condition1 of Def.1**.

- \(i_1 \in IMR1 \land i_2 \notin \text{ any IMR}\). For the same reason as above mentioned.

- \(i_1, i_2 \notin \text{ any IMR}\). This case conflicts with the **Theorem 1**.

2) Suppose that the structure relation between them is one of interaction relations in Group2 and the interaction pair is not the seed of any IMR, so there are three cases as follows.

- \(i_1 \in IMR1 \land i_2 \in IMR2\). IMR1 contains activity \(a\), but not activity \(c\). Since activity \(a, c\) make up Exclusive Split/Merge structure, therefore IMR1 only contains part of Exclusive Split/Merge structure, which conflicts with **Def.1** that the IMR consist of well-structured sub processes.

- \(i_1 \in IMR1 \land i_2 \notin \text{ any IMR}\). After correcting IMR1, the mismatched interaction pair \((i_1, i_2)\) is not corrected, which conflicts with the **Condition1 of Def.1**.

- \(i_1, i_2 \notin \text{ any IMR}\). This case conflicts with the **Theorem 1**.

As a result, **Theorem 2** is proved. \(\square\)

### 4.2.2. Identifying Algorithm for IMR

Based on the discovered CIMS between mismatched processes, all IMRs can be identified by the **Algorithm1** and **Algorithm2**.

**Algorithm 1**: Identifying all IMRs

**Input**: CIMS and business processes.

**Output**: All IMRs in the mismatched processes.

```plaintext
/*Store identified independent modifying region*/
IMRs={};
for each MIP in CMIS
  if MIP is seed of IMR
    /*Employ Algorithm 2 to identify an IMR by the seed.*/
    IMR =identifyIMRBySeed(MIP);
    IMRs.add(IMR);
  end if
end for

for each MIP in CMIS
  if( MIP(i1,i2) && i1 not in IMRs && i2 not in IMRs)
    IMR =identifyIMRBySeed(MIP);
    IMRs.add(IMR);
  end if
end for

for each IMR in IMRS
  mergeFlag=false;
  /*MR1,IMR2 contain common activities or transitions*/
  if ( IMR1 in IMRs && IMR2 in IMRs && hasCommonRegion(IMR1,IMR2))
    MergedIMR=merge(IMR1,IMR2);
    updateIMRs(MergedIMR);
    mergeFlag=true;
  end if
  /*If there are not any IMRs have common regions, then terminate this iteration */
  if(!mergeFlag)
    break;
  end if
end for
```

**Algorithm 2**: Identifying an IMR by the Seed

**Input**: A seed of an IMR

**Output**: IMR

Initially, the IMR is empty.

/*Find out minimal well-structured sub processes which contain activities engaged in the seed*/
SubProcess sub1=miniStructure(i1,i2, P);
SubProcess sub2=miniStructure(i1,i2, Q);
IMR.add(sub1);IMR.add(sub2);
IMR.add(i1);IMR.add(i2);
while ( i=a,b<>&a in IMR &b not in IMR){
**Theorem 3:** Correcting all the identified IMRs will make the business processes compatible.

**Proof:**
Assume that all the IMRs are corrected there is still a mismatched interaction pair \((i_1, i_2)\).

1. Assume that \(i_1, i_2\) are in the same IMR, the mismatched interaction pair has been corrected.
2. Assume that \(i_1, i_2\) are not in the same IMR, we prove that the assumption does not hold from the following three aspects, according to their structure relations.

   - **Group 1:** All the IMRs are corrected as an IMR, so they are in the same IMR.
   - **Group 2:** Assuming that \(i_1, i_2\) are in different IMRs, this assumption does not hold.
   - **Group 3:** If the IMRs are in the same IMR, then \(i_1, i_2\) are in the same IMR, but it conflicts with the assumption that interaction pair \((i_1, i_2)\) is mismatched. Therefore, correcting all the identified IMRs will make the business processes compatible.

### 4.3. IMR-based Correcting Plans Recommendation

This section discusses how to find out all the correcting plans with minimal cost evaluated by the size of modified structure relations. Although IMRs between mismatched business processes have been identified, using a breadth-first search algorithm to find out correcting plans for an IMR is still time-consuming, which is approximately exponential [1] to the size of mismatched interaction pairs in an IMR.

However, there are some patterns for IMRs. As mentioned in Sect. 3, there are two typical behavioral mismatches, which also be formalized as mismatch patterns in [2]. Although these patterns cannot cover all the cases when business processes are mismatched, these patterns are the basic units to combine mismatched business processes. Therefore, breadth-first search is employed to find out correcting plans for the two patterns in IMRs. But, whether there are other patterns and how to correct them will be discussed in our future work.

#### 4.3.2. Correcting Plans for Mismatch Pattern

This section discusses how to find out all the correcting plans with minimal cost evaluated by the size of modified structure relations. Although IMRs between mismatched business processes have been identified, using a breadth-first search algorithm to find out correcting plans for an IMR is still time-consuming, which is approximately exponential [1] to the size of mismatched interaction pairs in an IMR.

However, there are some patterns for IMRs. As mentioned in Sect. 3, there are two typical behavioral mismatches, which also be formalized as mismatch patterns in [2]. Although these patterns cannot cover all the cases when business processes are mismatched, these patterns are the basic units to combine mismatched business processes. Therefore, breadth-first search is employed to find out correcting plans for the two patterns in IMRs. But, whether there are other patterns and how to correct them will be discussed in our future work.

**Pattern 1:** There are a set of interactions \(I\) and an interaction \(i\). \(I\) and \(i\) make up Pattern 1 iff the interaction relations between interaction \(i\) and any interactions \(ix\) in the set \(I\) are all \(IR_{PQ}(i, ix) = \langle S_{Pre}, S_{Post} > \) or \(IR_{PQ}(i, ix) = \langle S_{Post}, S_{Pre} > \), and all the activities engaged in the interactions in the set \(I\) and interaction \(i\) in both processes can make up well-structured sub processes.

Employing breadth-first search algorithm will find out two correcting plans with minimal cost 1. (See Fig.5) Correcting Plan 1:

\[ IR_{PQ}(i3, i1) = \langle S_{Pre}, S_{Post} > \land \langle S_{Pre}, S_{Post} > \]

Correcting Plan 2:
Correcting Plan 1:
\[ IR_{i,j}(i,2) = \langle S_{Pre}, S_{Post} \rangle \rightarrow \langle S_{Post}, S_{Post} \rangle \]

Correcting Plan 2:
\[ IR_{i,j}(i,2) = \langle S_{Pre}, S_{Post} \rangle \rightarrow \langle S_{Post}, S_{Post} \rangle \]

Pattern 2: There is a set of interactions ISet. ISet makes up Pattern2 if any two interactions in ISet are one of the following interaction relations:
\[
IR_{i,j}(i,2) = \langle Com_E, S_{Pre} \rangle, \\
IR_{i,j}(i,2) = \langle Com_E, S_{Post} \rangle, \\
IR_{i,j}(i,2) = \langle S_{Pre}, Com_E \rangle, \\
IR_{i,j}(i,2) = \langle S_{Post}, Com_E \rangle, \\
IR_{i,j}(i,2) = \langle PL, Com_E \rangle \\
\]

and all the activities engaged in the interactions in ISet in both processes can make up well-structured sub processes.

Employing breadth-first search algorithm will find two correcting plans with minimal edit price \( n-1 \), while \( n \) indicates the size of ISet (See Fig.6). In order to provide multiple correcting plans for Pattern 2, it is reasonable to consider the interaction pairs marked by MIS* compatible under certain condition, as listed as follows. If the dataflow, or more concretely the condition of the choice, satisfying some constraints, these interactions probably will not cause business processes fail to interact properly. The business analyst is responsible for making the decision.

Correcting Plan 1:
\[ IR_{i,j}(i,2) = \langle Com_E, Com_E \rangle \]

Correcting Plan 2:
\[ IR_{i,j}(i,2) = \langle S_{Pre}, Com_E \rangle \rightarrow \langle Com_E, Com_E \rangle \\
IR_{i,j}(i,2) = \langle S_{Pre}, Com_E \rangle \rightarrow \langle Com_E, Com_E \rangle \\
\]

After finding out correcting plans for these patterns, the method of finding out correcting plans for IMRs is as follows: when employing breadth-first search to find out correcting plans for an IMR, use the pre-found correcting plans for the matched patterns. Furthermore the Branch and bound (BB) algorithm is used to reduce the size of search space, during finding correcting plans with the upper boundary of minimal cost (See Algorithm 3).

**Algorithm 3.** Finding Correcting Plans for an IMR

**Input:** IMR

**Output:** Multiple Correcting Plans

/* Store updated IMRs during search */

OpenList=\{IMR\};

/* Upper boundary of minimal cost */

MinnimalEP=\infty;

/* Store found correcting plans */

MinimalCPS=\{

for each IMR in OpenList

/* Setp1: Get correcting plans for each IMR */

IMR= OpenList.pop();

PreviousCorrectingPlan= getCPlanFor (IMR);

if(Pattern1 matched in IMR)

/* Get correcting plans for matched Pattern1 */

CorrectingPlans= getCPlanFor (Pattern1,IMR);

else if(Pattern2 matched in IMR)

/* Get correcting plans for matched Pattern2 */

CorrectingPlans= getCPlanFor (Pattern2,IMR);

else if(hasMismatchedInteractionPair(IMR))

/* Get correcting plans for all MIPs */

CorrectingPlans= getCPlanFor (IMR);

else /* Record found correcting plans */

if(PreviousCorrectingPlan.cost< MinnimalEP)

MinimalCPS.clear();

MinimalCPS.add(PreviousCorrectingPlan);

end if

if(PreviousCorrectingPlan.cost==MinnimalEP)

MinimalCPS.add(PreviousCorrectingPlan);

end if

end else

/*Step2: Update IMR according to correcting plans */

for each CorrectingPlan in CorrectingPlans

/* If edit price is smaller than the upper boundary, update IMR by the correcting plans */

if(CorrectingPlan. cost + PreviousCorrectingPlan.cost <= MinnimalEP &isFeasible(CorrectingPlan) )

IMR= updateIMR(CorrectingPlan);

OpenList.add(IMR);

end if

end for

In the worst case, the search space for the Algorithm 3 is equal with search space for classical breadth-first search algorithm, which is approximately exponential to the size of mismatched interaction pairs in an IMR. But the worst case occurs when the Pattern1 and Pattern2 are always not matched during correcting processes. Further-
thermore, as earlier mentioned, there are more than one minimal cost correcting plans for an IMR (See Sect. 4.1).

5. A Running Example

This section elaborates on the applicability of our approach by applying it to a running example and comparing the search space size with related technologies.

5.1. Correcting Plans for the Running Example

![Figure 7. Municipal Tax Collection](image)

An inter-organizational process model borrowed from [14] is modified to illustrate our approach. This example describes municipal tax collection, which involves two processes including a tax-payer (the Client) and a business agent (the Tax Advisor). Utilizing our previous work [4], a CIMS can be obtained which contains nine mismatched interaction pairs. Then two IMRs can be identified with the Algorithm1 (see Fig.7): IMR1 and IMR2. The IMR1 contains a mismatched interaction pair \((i_1, i_2)\).

Because \(IR_{C,T}(i_1, i_2) \neq S_{Post} \neq S_{Pre}\), while \(C, T\) indicate business process Client and Tax Advisor, Pattern1 is matched. The correcting plans for IMR1 are as follows.

Correcting Plan 1 for IMR1:

\[ IR_{C,T}(i_1, i_2) = S_{Pre} \neq S_{Post} \neq S_{Pre} \]

Correcting Plan2 for IMR1:

\[ IR_{C,T}(i_1, i_2) = S_{Pre} \neq S_{Post} \neq S_{Post} \]

The IMR2 contains two mismatched interaction pairs \((i_3, i_4)\) and \((i_3, i_5)\). Because

\[ IR_{C,T}(i_3, i_4) = PL, Com_E > \]

\[ IR_{C,T}(i_3, i_5) = PL, Com_E > \]

Pattern2 is matched. The correcting plans for IMR2 are as follows.

Correcting Plan 1 for IMR2:

\[ IR_{C,T}(i_3, i_4) = PL, Com_E > PL, PL > \]

\[ IR_{C,T}(i_3, i_5) = PL, Com_E > PL, PL > \]

Correcting Plan 2 for IMR2:

\[ IR_{C,T}(i_3, i_4) = PL, Com_E > PL, PL > \]

\[ IR_{C,T}(i_3, i_5) = PL, Com_E > PL, PL > \]

In a summary, there are four correcting plans for the running example, consist of different correcting plans for the two identified IMRs. As above mentioned, there are 9 mismatched interaction pairs in the running example, only correcting mismatched interaction pairs \((i_1, i_2), (i_3, i_4), (i_3, i_5)\) will make the business processes compatible.

Correcting Plan 1:

\[ IR_{C,T}(i_1, i_2) = S_{Pre} \neq S_{Post} \neq S_{Pre} \]

Correcting Plan 2:

\[ IR_{C,T}(i_3, i_4) = PL, Com_E > PL, PL > \]

Correcting Plan 3:

\[ IR_{C,T}(i_3, i_5) = PL, Com_E > PL, PL > \]

Correcting Plan 4:

\[ IR_{C,T}(i_3, i_5) = PL, Com_E > PL, PL > \]

However when Correcting Plan 3 and Correcting Plan 4 are adopted, the business analyst needs to ensure that both process Client and process Tax Advisor should satisfy certain conditional constraint (See Section 3).

5.2. Search Space Analysis

In the breadth-first search method, correcting plans are found without the use of the techniques of “IMR” and “Matching Patterns” proposed in the above. In order to show how much search space can be reduced by our approach in comparison with breadth-first search, we analyze the search space needed for the above running example.

1) There are 9 mismatched interaction pairs in the running example. The search space size of breadth-first search is exponential (See Section 4.3), which is \(O(a^n), a > 1\) for the running example.
2) There are two IMRs in the running example, the IMR1 contains a mismatched interaction pair, so the search space size for the IMR1 is $7^2$ (See Section 4.3). The IMR2 contains two mismatched interaction pairs. At the worst case, the search space size for the IMR2 is $2 \times 7^2$, when the two mismatched interaction pairs are corrected independently. As a result, the search space size of breadth-first search for IMRs without patterns is at most $56 \times 2$.

3) Pattern1 is matched in IMR1 and Pattern2 is matched in the IMR2 in the running example, so the search space size for the IMR1 is 2 and the search space size for the IMR2 is 2. As a result, the search space size for the mismatched business processes with our approach (See Sect.4.3) is 4.

<table>
<thead>
<tr>
<th>Table 2. Search Space Size Comparison.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Municipal Tax Collection</td>
</tr>
<tr>
<td>$O(a^2), a &gt; 1$</td>
</tr>
</tbody>
</table>

6. Conclusion

This paper aims at providing multiple minimal cost correcting plans for mismatched business processes to business modelers. A cost model is proposed to evaluate the correcting plans based on modified structure relations. In order to reduce the search space and avoid new generated mismatches during correcting processes, an independent modifying region-based method is proposed to obtain multiple minimal cost correcting plans. Correcting all the identified independent modifying regions (IMRs) will ensure business processes compatible. At last, a running example is given to illustrate the validity of our approach.

Future work will focus on the following open problems. Firstly, whether mismatched processes can always have correcting plans or not is not discussed in the paper. Secondly, the diversity of correcting plans is not well formalized and measured, which is a key factor for business modelers to make their preferable choices.

Acknowledgement

This work was supported in part by the National Basic Research Program of China (Grant: 2007CB310805), National Natural Science Fund of China (Grant: 60903137, 60873243), and National 863 Program (Grant: 2008AA01Z140). We appreciate anonymous reviewers for their valuable comments.

References