Automatic Detection Of Semantic Inconsistency Between BPMN Process Model and SBVR Rule Model

Akanksha Mishra

IIIT–Delhi (India)
Software Analytics Research Lab (SARL)
www.software-analytics.in
# MTECH Thesis Evaluation Committee Members

<table>
<thead>
<tr>
<th><strong>Thesis Adviser</strong></th>
</tr>
</thead>
</table>
| **Prof. Ashish Sureka**  
Principal Researcher at ABB Corporate Research Center (India)  
*Faculty In-charge, Software Analytics Research Lab (SARL)* |

<table>
<thead>
<tr>
<th><strong>External Examiner</strong></th>
</tr>
</thead>
</table>
| **Mr. Anjaneyulu Pasala**  
Principal Research Scientist at Infosys R&D Labs |

<table>
<thead>
<tr>
<th><strong>Internal Examiner</strong></th>
</tr>
</thead>
</table>
| **Prof. Somitra Kumar Sanadhya**  
Faculty Member at IIIT-Delhi |
Outline

1. Research Motivation and Aim
2. Related Work and Novel Research Contributions
3. Research Framework and Solution Approach
4. Experiment and Result
5. Limitations and Future Work
6. Conclusion
7. References
Business Process

- A collection of related, structured activities or tasks that produce a specific service or product for a particular customer.

- Organizations need process-modeling for various interests like
  - Business process integration and management
  - Evaluation and re-engineering of processes.
An Example: Taxi Order Process

Figure from [4]
A modeling notation that is understandable by various people of different expertise.

It covers large amount of real world concepts.

BPMN models provide the most of the formalized business knowledge, from which business vocabularies are to be extracted.

Other modeling-oriented standards, lack sufficient expressiveness compared to BPMN.
**Research Motivation and Aim**

**Problems with other Modeling Notations**

**Event-Driven Process Chain**

- Flowchart
- Ordered Graph of functions and events
- Connectors that allow alternative and parallel execution of processes

---

**Figure from [11]**
Unified Modeling Language

- Standardized modeling language enabling developers to **specify, visualize, construct and document** artifacts of software system.
- **Structural Diagrams** shows the things modeled in the system.
- **Behavioral Diagrams** shows what should happen in the system.

Definition from [12]
BPMN-SBVR

Research Motivation and Aim

An Example for BPMN Business Process Model

An Example for BPMN Business Process Model
Business Rule

- A rule that is under business jurisdiction.
- Intended to assert business structure or control or influence the behavior of the business.
- Define
  - Semantics of business concepts
  - Reactions to business events
  - Constraints and preconditions on tasks and activities
  - Rights and obligations of business actors.
An Example: Business Rule

Rule 1: In context of Order Entry System
➤ A customer must have a valid email address.

Rule 2: In context of License Inspection System
➤ A driver of a vehicle must have a valid Driver’s License.
A vocabulary that is under business jurisdiction.

It contains business data content that defines terms and other representations.

Set of specialized terms and definitions of concepts that a given organization or community uses in their talking and writing in the course of doing business.

Term “customer” in case of travel reservation scenario is an example of business vocabulary.
SBVR Business Vocabulary and Business Rule

- SBVR expresses business knowledge in a Controlled Natural Language (CNL).
- Purpose of using CNL is to ensure that representation of business concepts and rules is
  - unambiguous and understandable to humans as well as to computer systems.

Rules are built on Facts. Facts are built on Terms.
An Example for SBVR

- Term: customer
- Term: gold customer
- Term: itinerary
- Term: years
- Fact type: customer enters itinerary
- Fact type: gold customer is specification of customer
- Fact type: customer is years old
- Rule: It is necessary that each customer enters at least one itinerary.
- Rule: It is obligatory that each gold customer is specification of the customer.
- Rule: It is obligatory that each customer is at least 18 years old.
Inconsistency across Process and Rule Models

- **Inconsistency** — “Lacking in harmony between different parts or elements” or “One Part does not agree with other”

- **Syntactic Inconsistency**
  - Occurs when syntax are not followed properly

- **Semantic Inconsistency**
  - Occurs when IT System does not produce expected behavior

Figure from [15]
An Example – Syntactic Inconsistency

Syntax Error: if Sequence Flow was used

Syntax Error: if Parallel Gateway was used
An Example – Semantic Inconsistency

Task: Write a Program for addition of two numbers.

```c
#include<stdio.h>
void main(){
    int a, b, c;
    scanf(“%d”, &a);
    scanf(“%d”, &b);
    c = a*b;
    printf(“%d”, c);
}
```

No Syntax Error !! Working Fine.
But..
This program will not give expected output.

If a=4 and b = 4
Expected Output: a + b = 8
Output: a*b = 16
An Example - Semantic Inconsistency within BPMN-BPMN

**Taxi/Cab Booking**

**SCENARIO:** A family wants to book a cab for after an hour.

Number of seats and pick up time is not checked in any of BPMN diagram.
An Example - Semantic Inconsistency (SBVR-SBVR)

**Dinner Flow**

SCENARIO: Family wants to book a cab for after an hour.

**Rule Set 1**
- It is necessary that customer request for the cab.
- It is necessary that admin office review the request for the cab.
- It is necessary that garage allocate a cab if the request is approved.
- It is possible that family has more than 4 members.

**Rule Set 2**
- It is necessary that customer request for the cab.
- It is possible that customer rejects the booking.
- It is obligatory that travel agent assigns a cab operator if customer accepts the booking.
- It is necessary that cab driver pick up the customer.

No rule is defined for the seat availability for all members.
As per rule, customer is at least 18 years old to perform reservation. But there is no mention of age in BPMN diagram.
An Example - Semantic Inconsistency II

As per rule, each gold customer is eligible to the discount of 5% in the payment.

No mention of type of customer in this BPMN diagram.
To automate the process of detecting Semantic Inconsistency Violation across BPMN Process Model and SBVR Rule Model.

To extract Triplet from a English Sentence efficiently and accurately.

To develop system to detect faults within models which are human created artifacts.
To validate the developed system we conducted experiment on synthetic dataset related to many real world scenarios.
Outline

1. Research Motivation and Aim
2. Related Work and Novel Research Contributions
3. Research Framework and Solution Approach
4. Experiment and Result
5. Limitations and Future Work
6. Conclusion
7. References
Several studies dedicate to the integration of BPMN models and SBVR rules like [1], [2], [3], [5] etc.

Humm and Fengel have suggested assessing semantic consistency of business process models[6]

- Their solution follows EPC business process modeling language
- They propose rules for detecting syntactic and semantic inconsistencies within the business process model
Rusu et al[16] proposed an approach for the extraction of subject-predicate-object triplet from a sentence using four open-source parsers.

Dali et al[17] proposed machine learning approach to extract subject-predicate-object triplets from English sentences.

Zin Thu Thu Muint et al in [18] observed that other techniques developed are using one or the other parser affecting processing time for each sentences.
Ullmann[19] proposed the matching algorithm for the graph isomorphism and sub graph isomorphism.

- Backtracking procedure to reduce space search with effective look ahead function
- Independent on topology of graph

Another algorithm related to sub graph isomorphism is proposed by Cordella et al[20].

- Introduced unique representation for matching process known as SSR.
- Introduced set of feasibility rules
Cordella et al[21] proposed another work about the performance of the graph matching algorithm popularly known as VF Algorithm[20].

Done theoretical analysis of the algorithm about the computational complexity and memory requirements.
Cordella et al [22, 23] proposed an improved version of graph matching algorithm to solve isomorphism problems on attribute relational graphs.

- Focused on the semantic information of the graph
- Tried to organize search space in such a way to reduce memory requirements for large and medium sized graphs.
- Popularly known as VF2 Algorithm developed by reducing spatial complexity to $O(N)$ in best and worst case.
Novel Research Contributions

- Focus on the notion of “Model Quality Characteristic” i.e. consistency between BPMN Process Model and SBVR Rule Model.

- Used Grammatical Relations between words stored in the form of Typed Dependency to extract the Actor, Action and Object of a English Sentence.

- Proposed a synthetic dataset for the research problem in the integration of BPMN Process Model and SBVR Rule Model.
Novel Research Contributions

- Propose a novel Research Framework to detect instances of semantic inconsistency between BPMN Process Model and SBVR Rule Model.

- Demonstrate the effectiveness of our approach, discusses the strengths and limitations of our approach based on conducting experiments on synthetic dataset.
Outline

1. Research Motivation and Aim
2. Related Work and Novel Contributions
3. Research Framework and Solution Approach
4. Experiment and Result
5. Limitations and Future Work
6. Conclusion
7. References
Term 1: Isomorphism of Graphs

Two simple graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are isomorphic if there is a bijection (an one-to-one and onto function) $f$ from $V_1$ to $V_2$ with the property that $a$ and $b$ are adjacent in $G_1$ if and only if $f(a)$ and $f(b)$ are adjacent in $G_2$ for all $a$ and $b$ in $V_1$.

Isomorphism $f$ from the left to right graph is: $f(a) = e$, $f(b) = a$, $f(c) = b$, $f(d) = c$, $f(e) = d$.

Definition from [14]
Properties that two isomorphic graphs must both have:

- Same number of vertices
- Same number of edges
- Same degree of corresponding vertices
- Same number of connected components
- Same number of loops
- Same number of parallel edges

Two graphs that differ in any of these invariants are not isomorphic, but two graphs that match in all of them are not necessarily isomorphic.
Term 2: Adjacency Matrix

- Means to represent which vertices or nodes of a graph are adjacent to which other vertices or nodes.
- N X N zero-one matrix with 1 as its (i,j) entry when $v_i$ and $v_j$ are adjacent and 0 otherwise.

Definition from [14]
G1 and G2 are isomorphic if their vertices can be ordered in such a way that adjacency matrices $M_{G_1}$ and $M_{G_2}$ are identical.

However, it is easier to show two graphs are not isomorphic.
BPMN-SBVR Research Framework and Solution Approach

Architecture Diagram

BPMN Diagram

Generate

XML

Parse

Adjacency Matrix

Graph Isomorphism Algorithm

Semantic Repository

Vocabulary

Rules

Tagging & Parse

Syntax Tree

Typed Dependencies

Adjacency Matrix

Store As

Typed Dependencies
Solution Outline

- Generation of Adjacency Matrix for BPMN Process Model
- Graphical Representation of SBVR Rule Model
- Apply Graph Isomorphism Algorithm to graphical representation of BPMN Model and SBVR Model
  - If the graphs are isomorphic, then the models have no semantic inconsistency
  - Else, the models are semantically inconsistent.
**Proposed Algorithm I**

**Algorithm 1: XML based Adjacency Matrix**

**Data:** XML of BPMN Diagram

**Result:** Adjacency Matrix

1. Parse XML file and Extract Name and Id of the tags
2. Identify all the Activity in a process - these are nodes of the graph
3. Identify the Message Flow between pools and the Transitions within pools - these are edges of the graph
4. Adjacency Matrix has activities as the nodes of the matrix
5. Each entry in the matrix will be 1 if there is Message Flow or Transitions between two activities representing row and column. Otherwise corresponding entry will be 0.
BPMN Business Process Model for Travel Scenario

Message Flow

Transitions
<WorkflowProcess id=""/>

<Activities>
   <Activity Id="c150a5f4-aa70-48bd-845d-0a1f9204c21c" Name="Enter itinerary details into system"/>
   <Activity Id="bd1ede13-29dd-4e1e-b585-37377b67f63f" Name="Receive itinerary details from system"/>
   <Activity Id="593d8d29-54bd-4995-b8aa-b3f6e98cefc5" Name=""/>
   <Activity Id="b03cb8bd-1a0d-401f-aa07-493b29f0c2d9" Name="Request Vehicle Reservation">
</Activities>

........
........
ctd.
<?xml version="1.0" encoding="utf-8"?>

<Package>

<MessageFlows>
    <MessageFlow Id="1c9974a0-1a0d-411a-8f14-e2d43376deda"
        Source="c150a5f4-aa70-48bd-845d-0a1f9204c21c"
        Target="bd1ede13-29dd-4e1e-b585-37377b67f63f">''
    </MessageFlow>
</MessageFlows>

........
........
........

ctd.

Edges of the Graph
<Transitions>

<Transition Id="b2287b6a-b1f3-4785-8670-d07d507c1145"
   From="bd1ede13-29dd-4e1e-b585-37377b67f63f"
   To="593d8d29-54bd-4995-b8aa-b3f6e98cefc5">

<Transition Id="8dc77937-b3b7-4a64-b46f-43914f2fd85a"
   From="593d8d29-54bd-4995-b8aa-b3f6e98cefc5"
   To="b03cb8bd-1a0d-401f-aa07-493b29f0c2d9">

</WorkflowProcess>
</Package>

Edges of the Graph
### Adjacency Matrix

<table>
<thead>
<tr>
<th>ID</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>17</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**MessageFlow**

```
<MessageFlow Id="1c9974a0-1a0d-411a-8f14-e2d43376deda" Source="c150a5f4-aa70-48bd-845d-0a1f9204c21c" Target="bd1ede13-29dd-4e1e-b585-37377b67f63f"
```

**Transition**

```
<Transition Id="b2287b6a-b1f3-4785-8670-d07d507c1145" From="bd1ede13-29dd-4e1e-b585-37377b67f63f" To="593d8d29-54bd-4995-b8aa-b3f6e98cefc5">
```

```
<Transition Id="8dc77937-b3b7-4a64-b46f-43914f2fd85a" From="593d8d29-54bd-4995-b8aa-b3f6e98cefc5" To="b03cb8bd-1a0d-401f-aa07-493b29f0c2d9">
```
Research Framework and Solution Approach

BPMN Graph
Proposed Algorithm II

Algorithm 2: Typed Dependency based Triplet Extraction

Data: Rule written in Natural Language
Result: Triplet Extraction from Sentences

1. foreach Rule in File do
   2. foreach Object in List do
      3. if DependencyName not equals "mark" then
         4. if DependencyName equals "root" then
             5. add Node String to Verb
         6. if DependencyName equals "nsubj" or "nsubjpass" then
             7. if DependencyName preceeded by "compound" then
                 8. Concatenate compound and Node String
                 9. add Node String to Subject
             else
                10. add Node String to Subject
         11. if DependencyName equals "dobj" then
             12. if DependencyName preceeded by "compound" then
                 13. Concatenate compound and Node String
                 14. add Node String to Object
             else
                15. add Node String to Object
Proposed Algorithm II contd.

```
18       else
19        if DependencyName equals "nsubj" then
20               add subject governor to SubjectGoverner
21                     if DependencyName preceded by "compound" then
22                                    Concatenate compound and Node String
23                                    add Node String to Subject
24                      else
25                               add Node String to Subject
26                    if DependencyName equals "dobj" then
27                         add object governor to ObjectGoverner
28                           if SubjectGoverner equals ObjectGoverner then
29                                         add ObjectGoverner to Verb
30                             if DependencyName preceded by "compound" then
31                                      Concatenate compound and Node String
32                                      add Node String to Object
33                                else
34                                      add Node String to Object
```
It is necessary that each customer enters at least one itinerary.
Algorithm 3: Triplet based Rule Classification

- **Data:** Actor-Action-Object of a rule
- **Result:** Classification of a rule

1. Read a file
2. **foreach** Rule in File do
3. Create an arraylist for an Actor, an Object and an Action
4. **if** Actor != NULL and Verb != NULL and Object != NULL **then**
   5. add Actor, Verb and Object to arraylist
   6. Transitive Verb
   7. Rule participate in Business Process
5. **else**
6. Intransitive Verb
7. Rule do not participate in Business Process
It is necessary that each customer enters at least one itinerary.

Universal dependencies:

- Subject: det(customer-2, each-1), nsubj(enter-3, customer-2)
- Verb: root(ROOT-0, enter-3), case(least-5, at-4)
- Direct Object: nmod:npmod(one-6, least-5), nummod(itinerary-7, one-6), dobj(enter-3, itinerary-7)
- Transitive Verb: participate in Business Process
It is necessary that each customer is at least 18 years old.
### Triplet (S-V-O)

<table>
<thead>
<tr>
<th>Actor</th>
<th>Action</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer</td>
<td>Enters</td>
<td>Itinerary</td>
</tr>
<tr>
<td>Itinerary</td>
<td>Has</td>
<td>Reservation request</td>
</tr>
<tr>
<td>Booking system</td>
<td>Contacts</td>
<td>Reservation system</td>
</tr>
<tr>
<td>Itinerary</td>
<td>Has</td>
<td>Reservation request</td>
</tr>
<tr>
<td>Reservation system</td>
<td>Confirms</td>
<td>Reservation request</td>
</tr>
<tr>
<td>Booking system</td>
<td>Debits</td>
<td>Credit card</td>
</tr>
<tr>
<td>Booking system</td>
<td>Receives</td>
<td>Failure message</td>
</tr>
<tr>
<td>Customer</td>
<td>Revises</td>
<td>Itinerary</td>
</tr>
</tbody>
</table>
SBVR Graph

- Customer
- Itinerary
- Reservation request
- Reservation system
- Booking system
- Credit card
- Failure message
Proposed Algorithm IV (VF2 Algorithm)

Algorithm 4: PROCEDURE Match(s)

Data: an intermediate state $s$; the initial state $s_0$ has $M(s_0) = \emptyset$

Result: the mappings between the two graphs

1. if $M(s)$ covers all nodes of $G_2$ then
2. OUTPUT $M(s)$
3. else
4. Compute the set $P(s)$ of the pairs candidate for inclusion in $M(s)$
5. foreach $p$ in $P(s)$ do
6. if the feasibility rules succeed for the inclusion of $p$ in $M(s)$ then
7. Compute the state $s'$ obtained by adding $p$ to $M(s)$
8. CALL Match($s'$)
9. Restore Data Structures
10. END PROCEDURE Match
VF2 Algorithm Procedure Match(s)

Graph V

Graph V'
**Algorithm 5: Subgraph-Subgraph Mapping Algorithm**

**Data:** Two Adjacency Lists  

**Result:** mapping between two graph which are isomorphic

1. Create a Graph $G_1$ by reading file containing adjacency list of first graph  
2. Create a Graph $G_2$ by reading file containing adjacency list of second graph  
3. Find node-induced subgraph of all possible combination of length of nodes of $G_1$  
4. **foreach** subgraph $G'$ of $G_1$ **do**  
   5. Find subgraph of all possible combination of length of nodes of $G_2$  
5. **foreach** subgraph $G''$ of $G_2$ **do**  
   6. Perform Matching between subgraph $G'$ and $G''$  
   7. **if** $(G',G'')$ are isomorphic **then**  
   8. Print Mapping between $(G',G'')$
Outline

1. Research Motivation and Aim
2. Related Work and Novel Contributions
3. Research Framework and Solution Approach
4. Experiment and Result
5. Limitations and Future Work
6. Conclusion
7. References
## Synthetic Dataset

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Scenario</th>
<th>Consistent</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Travel Reservation</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>Cab Booking</td>
<td>Yes</td>
</tr>
<tr>
<td>3</td>
<td>Employee Reimbursement Request</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>Order Fulfillment</td>
<td>No</td>
</tr>
</tbody>
</table>
Consider a cab booking scenario where:
- Customer request for the cab booking.
- Travel agent get the request and check the availability of the cab.
- Inform the customer about the booking status of cab and he/she can accept or reject depending on the booking details.
- If customer confirms the cab booking, then travel assign will assign a driver for the cab.
- Cab driver picks up the customer and drops him or her at the scheduled destination.
BPMN Business Process Model for Cab Booking

Customer
- Enter Itinerary details
- Request Booking
- Get Booking Request
- Check Availability
- Get Alternative Time
- Propose Booking Status
- Get Booking Status
- See Booking Details
- Confirm Booking

Travel agent
- Yes
- Accepted
- Not Accepted
- Assign Cab Driver
- Pickup and Drop Customer
- End

Cab driver
### Consistent Scenario: Cab Booking

#### Activity (Nodes)

<table>
<thead>
<tr>
<th>Activity ID</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>839b6582-dfb0-4c74-9115-999630225964</td>
<td>1</td>
</tr>
<tr>
<td>1f142692-4b9c-42bd-9507-6b0f1b2a245c</td>
<td>2</td>
</tr>
<tr>
<td>587a2fb7-58be-49f9-8fdc-04d28eb9b161</td>
<td>3</td>
</tr>
<tr>
<td>9ca0e033-0d04-467e-a3cc-41f652981cb3</td>
<td>4</td>
</tr>
<tr>
<td>6e92b4d3-a87a-41d6-82a8-48eb107dba58</td>
<td>5</td>
</tr>
<tr>
<td>8f71ea67-d2d8-45d4-9082-c4bed32e3cfc</td>
<td>6</td>
</tr>
<tr>
<td>d1625337-5235-4fa3-8a31-22bbcc16ce9e</td>
<td>7</td>
</tr>
<tr>
<td>5d116fee-2c01-4616-8f24-64b51352be7a</td>
<td>8</td>
</tr>
<tr>
<td>733b1c38-a17d-4349-b814-1cab5c914336</td>
<td>9</td>
</tr>
<tr>
<td>568469a9-3e2c-4a86-8dde-7c678369c3c2</td>
<td>10</td>
</tr>
<tr>
<td>139271c9-9b7c-4728-806f-60fdd21e2232</td>
<td>11</td>
</tr>
<tr>
<td>52fc9249-c95c-4171-bc07-d57d812b05f3</td>
<td>12</td>
</tr>
<tr>
<td>52d843ba-623f-4d5f-8c7f-d2bc79431911</td>
<td>13</td>
</tr>
<tr>
<td>e0c4df79-0d6a-40df-8489-531939ae4f26</td>
<td>14</td>
</tr>
<tr>
<td>a8ae54ee-d6eb-4ccb-9219-908127c7fdb</td>
<td>15</td>
</tr>
</tbody>
</table>
### Consistent Scenario: Cab Booking

#### Message Flow & Transition (Edges)

<table>
<thead>
<tr>
<th>FROM</th>
<th>TO</th>
</tr>
</thead>
<tbody>
<tr>
<td>52fc9249-c95c-4171-bc07-d57d812b05f3</td>
<td>e0c4df79-0d6a-40df-8489-531939ae4f26</td>
</tr>
<tr>
<td>587a2fb7-58be-49f9-8fdc-04d28eb9b161</td>
<td>9ca0e033-0d04-467e-a3cc-41f652981cb3</td>
</tr>
<tr>
<td>9ca0e033-0d04-467e-a3cc-41f652981cb3</td>
<td>568469a9-3e2c-4a86-8dde-7c678369c3c2</td>
</tr>
<tr>
<td>139271c9-9b7c-4728-806f-60fdd21e2232</td>
<td>52d843ba-623f-4d5f-8c7f-d2bc79431911</td>
</tr>
<tr>
<td>6e92b4d3-a87a-41d6-82a8-48eb107dba58</td>
<td>8f71ea67-d2d8-45d4-9082-c4bed32e3cfc</td>
</tr>
<tr>
<td>1f142692-4b9c-42bd-9507-6b0f1b2a245c</td>
<td>6e92b4d3-a87a-41d6-82a8-48eb107dba58</td>
</tr>
<tr>
<td>e0c4df79-0d6a-40df-8489-531939ae4f26</td>
<td>a8ae54ee-d6eb-4ccb-9219-908127c7fdba</td>
</tr>
<tr>
<td>5d116fee-2c01-4616-8f24-64b51352be7a</td>
<td>587a2fb7-58be-49f9-8fdc-04d28eb9b161</td>
</tr>
<tr>
<td>8f71ea67-d2d8-45d4-9082-c4bed32e3cfc</td>
<td>d1625337-5235-4fa3-8a31-22bbcc16ce9e</td>
</tr>
<tr>
<td>d1625337-5235-4fa3-8a31-22bbcc16ce9e</td>
<td>733b1c38-a17d-4349-b814-1cab5c914336</td>
</tr>
<tr>
<td>568469a9-3e2c-4a86-8dde-7c678369c3c2</td>
<td>139271c9-9b7c-4728-806f-60fdd21e2232</td>
</tr>
<tr>
<td>839b6582-dfb0-4c74-9115-999630225964</td>
<td>1f142692-4b9c-42bd-9507-6b0f1b2a245c</td>
</tr>
</tbody>
</table>
### Message Flow & Transition (Edges)

<table>
<thead>
<tr>
<th>FROM</th>
<th>TO</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>14</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>11</td>
<td>13</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>14</td>
<td>15</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>
BPMN-SBVR

Experiment and Result

Consistent Scenario: Cab Booking

BPMN Graph

1 → 2 → 5 → 6 → 7 → 9
8 → 3 → 4 → 10 → 11 → 13
12 → 14 → 15
It is necessary that customer request the cab booking.

It is necessary that the travel agent checks availability of the cab.

It is necessary that the travel agent provides booking information to the customer if the cab is available.

It is possible that the customer cancels the cab booking.

It is necessary that the travel agent assigns the cab driver if the cab is available and the customer confirms the cab booking.

It is necessary that the cab driver picks up the customer at the scheduled time and drops the customer at destination. and many more....
### Triplet (S-V-O)

<table>
<thead>
<tr>
<th>Actor</th>
<th>Action</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer</td>
<td>enters</td>
<td>Itinerary details</td>
</tr>
<tr>
<td>Customer</td>
<td>Request</td>
<td>Cab booking</td>
</tr>
<tr>
<td>Travel agent</td>
<td>Checks</td>
<td>Availability</td>
</tr>
<tr>
<td>Travel agent</td>
<td>Provides</td>
<td>information</td>
</tr>
<tr>
<td>Customer</td>
<td>Confirms</td>
<td>Cab booking</td>
</tr>
<tr>
<td>Customer</td>
<td>Cancels</td>
<td>Cab booking</td>
</tr>
<tr>
<td>Travel agent</td>
<td>Assigns</td>
<td>Cab driver</td>
</tr>
<tr>
<td>Customer</td>
<td>Confirms</td>
<td>Cab booking</td>
</tr>
<tr>
<td>Cab driver</td>
<td>picks</td>
<td>customer</td>
</tr>
</tbody>
</table>
The image contains a SBVR graph illustrating a consistent scenario for a cab booking process. The graph shows the following entities and their relationships:

- **Travel Agent**
- **Information**
- **Cab Driver**
- **Customer**
- **Cab Booking**
- **Itinerary Details**
- **Availability**

The graph shows the interactions and dependencies between these entities, indicating how information flows and decisions are made in the cab booking process.
VF2 Algorithm (Graph - Subgraph)

- Directed Graph G1: BPMN
  - Nodes: 15
  - Edges: 12
- Directed Graph G2: SBVR
  - Nodes: 7
  - Edges: 9
- G1 is not isomorphic to G2
- G1 is not subgraph isomorphic to G2
- Mapping between nodes:
  - {}
## Experimental Result

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Expected</th>
<th>BPMN Nodes (G1)</th>
<th>SBVR Nodes (G2)</th>
<th>Subgraph of BPMN (G3)</th>
<th>Subisomorphic (G2,G3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Travel Reservation</td>
<td>Inconsistent</td>
<td>17</td>
<td>10</td>
<td>7</td>
<td>True</td>
</tr>
<tr>
<td>Cab Booking</td>
<td>Consistent</td>
<td>15</td>
<td>7</td>
<td>5</td>
<td>True</td>
</tr>
<tr>
<td>Employee Reimbursement</td>
<td>Consistent</td>
<td>13</td>
<td>5</td>
<td>3</td>
<td>True</td>
</tr>
<tr>
<td>Request</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Order Fulfillment</td>
<td>Inconsistent</td>
<td>10</td>
<td>7</td>
<td>4</td>
<td>True</td>
</tr>
</tbody>
</table>
Outline

1. Research Motivation and Aim
2. Related Work and Novel Contributions
3. Research Framework and Solution Approach
4. Experiment and Result
5. Limitations and Future Work
6. Conclusion
7. References
Limitations and Future Work

- Experiments performed are conducted on a very small synthetic dataset.
- Future work involves performing experiments with the **real world dataset**.
- No feedback or involvement of industry people in the approach.
- Future work involves exposing our work to **industrial environment**.
Limitations and Future Work

- Triplet (Subject-Verb-Object) Extraction is limited to simple sentences.
- Improve algorithm to handle complex and compound sentences.

- Used Typed Dependency of parse tree generated using Stanford NLP Parser.
- In future, Combine NLP with other machine learning techniques to increase the confidence in results.
Experimental results indicates that it is not possible to detect instances of inconsistency between BPMN process model and SBVR rule model.

In future, Try to improve results by conducting experiments on real world dataset and working on the proposed framework.
Proposed a novel approach for the detection of instances of semantic inconsistency between BPMN process model and SBVR Rule Model.

Key Components of framework are:

- Tags extraction from xml
- Triplet extraction of action-oriented rules using grammatical relations between words
- Determination of node-induced subgraph and apply VF2 algorithm
- Created a synthetic dataset
References


Business Process BPMN Diagrams
Habich, D. et al.

Bernhard G. Humm and Janina Fengel. 
International Conference on Business Information Systems (ICBIS 2012)

Tomas Skersys, Rimantas Butleris and Kestutis Kapocius. 
Extracting business vocabularies from business process models : SBVR and BPMN standards-based approach. 
AIP Conference Proceedings

First Class Computer Consulting 

Business Process 
http://en.wikipedia.org/wiki/Business_process
References-III

Anomalies
http://www.solutionsim.com/packages/

Event Process Chain
http://www.processorientation.com/wp-content/uploads/EPC_example.png

Unified Modeling Language
http://www.techopedia.com/

Business Rule Image

Graph Isomorphism and Adjacency Matrix Definition
Based on Slides by Y.Peng (University of Maryland)

Inconsistency Image
http://www.monpa.com/wcp/creativecards.html
Delia Rusu, Lorand Dali, Blaz Fortuna, Marko Grobelnik and Dunja Mladenic.
Triplet Extraction from sentences.
*In proceedings of 10th International Multiconference, pp. 8-12, 2007.*

Lorand Dali and Blaz Fortuna.
Triplet extraction from sentences using svm.
*Proceedings of SiKDD, 2008.*

Zin Thu Thu Myint and Kay Khaing Win.
Triplet Patterns Extraction for Accessing Data on Ontology.

J R Ullman.
An Algorithm for Subgraph Isomorphism.
*ACM, 1976.*


