Prozessvarianten in BPMN

Inhalt

[Einsatzmöglichkeiten – Wozu eigentlich das Ganze? 1](#_Toc240954379)

[Eingesetzte Techniken – Wie macht man’s am besten? 2](#_Toc240954380)

[Label Matching – Heißt du so ähnlich? 2](#_Toc240954381)

[Syntatic Matching 2](#_Toc240954382)

[Semantic Matching 2](#_Toc240954383)

[Attribute Matching – Bist du so ähnlich? 3](#_Toc240954384)

[Context Matching – Mit wem hast du zu tun? 3](#_Toc240954385)

[Structural Matching – Zum Wohle des Ganzen 4](#_Toc240954386)

[Graph Edit Distance - Wann sind zwei Graphen ähnlich? 4](#_Toc240954387)

[Grenzen – Was geht nicht? 5](#_Toc240954388)

[Ausblick – Was kann man noch so alles machen? 5](#_Toc240954389)

# Einsatzmöglichkeiten – Wozu eigentlich das Ganze?

* Vergleich unterschiedlicher Versionen eines Prozesses ohne Log und IDs… auch zwischen verschiedenen Tools(wenn importierbar) und mit alten Prozessen(abwärtskompatibel) – das können z.B. Aris oder Oracle BPA nicht…
* Prozesse unterliegen einem ständigen Wandel in Unternehmen – mit dem hier benutzten Verfahren können zum Beispiel die Änderungen durch Restrukturierungen nachvollzogen werden – auch wenn inzwischen ein anderes Tool für die Prozessverwaltung genutzt wird
* Es existieren verschiedene Varianten eines Prozesses
* Vergleich von Referenzprozess und der tatsächlichen Implementierung
* Prozesse unterschiedlicher Abteilungen vergleichen
  + was können sie voneinander lernen
  + welche Teile können zusammengelegt werden 🡪 Kosteneinsparung
  + …
* Bei Mergern von Unternehmen – Wie können wir unsere Prozesse am besten miteinander kombinieren?
* Ganz nebenbei wird auch bestimmt wie ähnlich zwei Prozesse sind, was z.B. für (semantische) Suchen in Prozessrepositories oder Features wie „zeige ähnliche Prozesse“ oder der Vermeidung von Duplikaten von Bedeutung ist

# Eingesetzte Techniken – Wie macht man’s am besten?

## Label Matching – Heißt du so ähnlich?

### Syntatic Matching

#### Preprocessing

* Groß/Kleinschreibung
* Häufig vorkommende Wörter ( a, the, is ….) entfernen
* Sonderzeichen entfernen
* Wörter auf Wortstamm zurück führen (z.B. Porter‘s Stemming Algorithmus)
* Überflüssige Leerzeichen entfernen

#### Damerau–Levenshtein Distanz

Alternative: Hammingway-Distanz

### Semantic Matching

Es wird Wordnet mit der Rita-Schnittstelle genutzt, um Synonyme zu finden. Es wird kein direktes Matching durchgeführt, es wird lediglich überprüft, ob sich ein Synonym eines Wortes im anderen Label befindet. Dadurch kann Rechenzeit gespart werden, da Label nur wenige Wörter umfassen und so eine Wort-zu-Wort Zuordnung überflüssig ist.

Semantic Matching erweist sich als der Teil des Vergleichs, der am meisten Rechenzeit benötigt, hier könnte durch das Laden von Wordnet in den RAM Abhilfe geschaffen werden.

Es wäre auch möglich semantisches und syntaktisches Matching zu integrieren. So könnte man Wort für Wort die Syntaktische Ähnlichkeit berechnen und unterhalb einer bestimmten Ähnlichkeit ein Synonym suchen. Momentan werden syntaktische und semantische Ähnlichkeit separat berechnet und danach gewichtet addiert.

## Attribute Matching – Bist du so ähnlich?

Es werden alle Attribute genutzt, die die Bedeutung des BPMN Diagrammes bestimmen, d.h. die Farbe fließt zum Beispiel nicht mit ein. Dagegen sind der Typ(z.B. sendendes intermediate-Event) oder Schleifen von Bedeutung.

Bei der Kategorisierung der Typen entstehen neben Kanten vier Äquivalenzklassen: Gateways, Events, Activities und Cluster.

**Gateways** können nur untereinander verglichen werden. Zwei parallele Gateways sind ähnlicher als ein paralleles und ein Inklusives. Der dadurch entstehende Unterschied (auch bei den anderen Typen Events, Activities und Cluster) in der Attributsähnlichkeit wird durch die entsprechenden Konstanten bestimmt.

**Events** können momentan nur untereinander verglichen werden, es ist aber auch möglich, diese (mit entsprechender Verringerung der Ähnlichkeit) in bestimmten Fällen mit Activities vergleichbar zu machen. Startereignisse sollten nicht mit Endereignissen vergleichbar sein, da dies nur bei einer Spiegelung des Graphen sinnvoll wäre. (Anmerkung: die Optimierung vor dem strukturellen Matching und das Attributmatching zeigen hier noch unterschiedliches Verhalten) Bei Prozessen scheint dies aber kein Use-Case zu sein. Startereignisse sind z.B. mit einer kleinen Strafe mit anderen Typen von Startereignissen vergleichbar und mit einer größeren auch mit Zwischenereignisssen.

**Activities** können mit Activities, Clustern(außer Pool) und bedingt mit Ereignissen verglichen werden.

**Cluster** für Cluster gilt dann entsprechendes.

Attribute Matching ist BPMN-spezifisch und muss angepasst werden, wenn andere Prozessmodelle genutzt werde. Ebenso ist eine Optimierung vor dem Structural Matching, um die Paare bei denen ein Matching keinen Sinn macht auszufiltern, BPMN-spezifisch. Darüber hinaus muss auch die Ausgabe(mergeModel) angepasst werden.

## Context Matching – Mit wem hast du zu tun?

Vergleiche die Vorbereiche und Nachbereiche der Knoten, d.h. die direkt nachfolgenden/vorgehenden Knoten und Kanten. Erstelle dazu ein Mapping zwischen den jeweiligen Vorbereichen/Nachbereichen. Nutze dann die Ähnlichkeit der Mappings, um die Ähnlichkeit des Knoten zu modifizieren. Rekursionen und Iterationen können möglicherweise zur Steigerung der Genauigkeit genutzt werden.

## Behavior Matching – Wen kannst du erreichen, von wem wirst du erreicht?

Bestimme für jeden Knoten, welche Knoten von ihm erreicht werden können und von welchen er erreicht kann.

## Structural Matching – Zum Wohle des Ganzen

### Graph Edit Distance - Wann sind zwei Graphen ähnlich?

#### Auswahl des Algorithmus

#### N-to-M Matching – die einfache Variante

* Mittels cut-off value
* Es wird einfach jedes/das erste Paar zum Mapping hinzugefügt, dass eine bestimmte Mindestähnlichkeit hat
* Erzeugt schlechtes Mapping – sollte nicht verwendet werden

#### Greedy Graph Matching – sparsam aber nur eine Näherung

* Solange die Graph Edit Distance sinkt, füge das Paar aus der Menge der offenen Paare hinzu, dass die Graph Edit Distance am meisten verringert
* Entferne alle Paare aus der Menge der offenen Paare, die eines der beiden Elemente des eingefügten Paares enthält
* Zeit: O(n³) {n ist die Zahl der Knoten des größeren Graphen}
* Raum: O(n² )

#### A-Star Graph Matching – optimales Mapping, aber hohe Speichkomplexität

* Initialisiere die Menge aller Mappings: Jedes Mapping wird durch ein mögliches Paar repräsentiert
* Wähle das Mapping mit der kleinsten Graph Edit Distance und füge ihm das Paar hinzu, dass seine Graph Edit Distance am meisten verbessert, sollten bereits alle Elemente in diesem Mapping vorhanden sein und so kein mögliches Paar existieren, ist dies das optimale Mapping
* Beginne von vorne
* Zeit: O(n²\*m) {n ist die Zahl der Knoten des ersten Graphen, m die Zahl der Knoten im zweiten Graphen}
* Raum: O( )

#### Optimierung

* als offene Paare nur die Paare betrachten, bei denen ein Vergleich auch sinnvoll ist (Event und Gateway z.B. nicht)
* Ähnlichkeiten der Paare cachen
* Einen Index auf die beiden Elemente eines Paares und die beiden Elemente zusammen erstellen
* Paare, die unterhalb einer bestimmten Ähnlichkeit liegen, nicht in offene Paare einfließen lassen

# Grenzen – Was geht nicht?

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Algorithmus [[1]](#footnote-1) | wasim | wfmate | wfmatn | Präzision | Benötigte Zeit |
| Greedy | 0,9 | 0,4 | 0.1 | 0,84 | 3,8 Sek. |
| Exhaustive | 0,8 | 0,2 | 0.1 | 0,82 | 53,7 Sek. |
| Process Heuristic | 0,8 | 0,2 | 0.1 | 0,83 | 14,2 Sek. |
| A-star | 0,1 | 0,7 | 0.2 | 0,86 | 15,7 Sek. |

* asim: Gewicht für die Durchschnittliche Ähnlichkeit der zugeordneten Knoten
* fmate: Gewicht für den Anteil der ersetzten Kanten
* fmatn: Gewicht für den Anteil der ersetzten Knoten

Besonders bei sehr unterschiedlichen Prozessen, wie z.B. Referenzprozesse und deren tatsächliche Implantierung und Prozessen aus unterschiedlichen Unternehmen kann die die Präzision gering sein (60%), da sich bei sehr Unterschiedlichen Graphen auch Fehler durch Kontext- und Verhaltensanalyse fortsetzen. Daher stellt der Prozessvergleich (besonders) dort lediglich eine Hilfe für den Menschen dar. Bei unterschiedlichen Versionen desselben Prozesses (mit gleichen IDs) ist die Erfolgsquote dagegen nahe 100% und kann als verlässlich angesehen werden.

# Ausblick – Was kann man noch so alles machen?

* Verfeinerungen in Unterscheidung der Klassen
* Andere Darstelllungsmöglichkeiten z.B. beide Graphen untereinander, Übereinstimmungen verbinden, Anzeigen der Ausgangsgraphen mit entsprechender Farbmarkierung
* Performance…
* Stemming
* Optimieren des Codes auf Lesbarkeit
* Verbesserung des semantischen Matchings, Unterstützung von Deutsch neben Englisch
* Übersetzungen Deutsch – Englisch einbeziehen
* Beschleunigung von Wordnet (komplett in RAM laden)
* Nutzung von Pools im Context Matching
* Matching von Unterprozessen
* …

1. Abgeleitet aus : Remco Dijkman, Marlon Dumas, Luciano Garca-Banuelos: Graph Matching Algorithms for BusinessProcess Model Similarity Search [↑](#footnote-ref-1)