Assisting Business Process Design by Activity Neighborhood Context Matching

Nguyen Ngoc Chan, Walid Gaaloul, and Samir Tata

Information Department, TELECOM SudParis
UMR 5157 CNRS Samovar, France

Abstract. Speeding up the business process design phase is a crucial challenge in recent years. Some solutions, such as defining and using reference process models or searching similar processes to a working one, can facilitate the designer's work. However, recommending the whole process can make the designer confused, especially in case of large-size business processes. In this paper, we introduce the concept of activity neighborhood context in order to propose an approach that fasten the design phase regardless the size of business process. Concretely, we recommend the designer the activities that are close to the designing process from existing business processes. We evaluate our approach on a large collection of public business processes. Experimental results show that our approach is feasible and efficient.

1 Introduction

The advantages of business process design have involved many industrial and research contributions on facilitating the business process design, which is the initial and key step that impacts the completeness and success of a business process. In this paper, we present an original approach to help to facilitate the design phase by recommending business process designers a list of relevant activities to the ongoing designed process. Consider a scenario where a business process designer is designing a "train-reservation" process to provide a booking service (Fig. 1): whenever the train operating company receives a reservation request, it searches trains according to the request details, presents possible alternatives to the customer and waits for a response. If it receives a cancel request, the process will be terminated; otherwise, it will ask the customer for the credit card information, then process the payment and send back the customer the reservation confirmation with the payment details.

The "train-reservation" process in Fig. 1 can achieve the required business goal. However, the process design could not stop at that point as the preliminary design requirements could evolve. He might want to: (i) add new functionalities in the preliminary process, (ii) design a new variant of the preliminary process respecting new business constraints or contexts, or (iii) find alternative activities in order to better handle activity failure or exception.

To help the designer achieve his goals, instead of recommending business processes, we propose to recommend activities that have similar neighborhood context with a selected one. This context is defined as a business process fragment

C. Liu et al. (Eds.): ICSOC 2012, LNCS 7636, pp. 541–549, 2012.

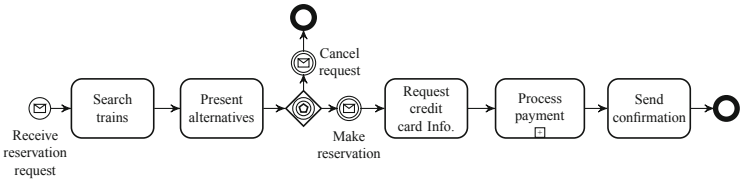


Fig. 1. Train reservation process

around an activity, including the associated activity and connection flows connecting it and its neighbors. For a selected activity, we match its neighborhood context with the neighborhood contexts of other activities. A matching between two neighborhood contexts is scored by a similarity value. Then, base on the similarity values, we present for the business designer N activities that have highest similarity values.

For example, if the designer selects activities: "Search trains", "Request credit card Info." and "Process payment" for recommendations, our approach recommends him relevant activities as given in Fig. 2.

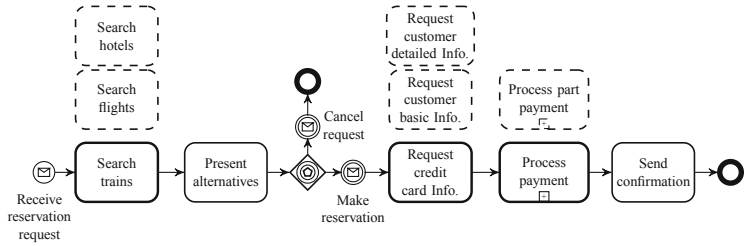


Fig. 2. Recommendations for the train reservation process

The recommendations given by our approach do not make the designer confused since they do not recommend the whole business structure. In contrast, short lists of recommended activities can help the designer easily open his view to improve the working process. For example, those recommendations, the designer is supposed to have ideas to improve the "train-reservation" process by such ways that: he can either add the "Request customer basic Info." activity for future customer services or improve the current process to achieve a traveling service, which combines activities in the "train-reservation" and "hotel-reservation" processes (Fig. 3).

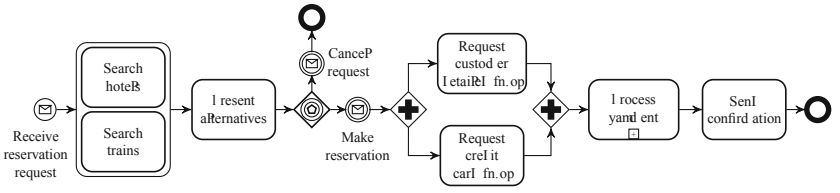


Fig. 3. New traveling process improved from the train reservation process

This paper is organized as following: the next section presents the related work. Details of the approach are elaborated in section 3. Section 4 shows our implementation and experiments. Finally, we conclude our work in section 5.

2 Related Work

Some existing approaches [1,2,3] target to fasten the design phase by retrieving similar process to the current designed process from repositories. They proposed either to rank existing business process models for similarity search [1,4], or to measure the similarity between them [2,3,5] for creating new process models. In our approach, we focus partially on the business process and take into account only the activity neighborhood context for recommendations instead of matching the whole business process.

R. Dijkman et. al. [6] used Levenshtein distance to compare the activity labels; graph edit distance and vector space model to determine the similarity between business process structures. They also proposed the ICoP framework [7] to identify the match between parts of process models using these metrics. Different from them, we focused on activity neighborhood contexts with layers and zones. We compute the similarity between neighborhood contexts based on the matching of connection flows in zones with zone weight consideration instead of matching activity labels or matching virtual documents.

S. Sakr et. al. [8] proposed a query language which takes into account the partial process models to manage business process variants. They, however, retrieve parts of processes based on strictly mapping to a structured input without considering the activity similarity. In our work, we retrieve the relevant activities based on the similarity values which are computed based on a tree structure mapping (section 3.2).

A search framework that aims at retrieving process segments was proposed by M. Lincoln et. al. [9]. In their work, they defined the object grouping model (OGM) which includes the relationship between a primary object and others in a process segment. Different from them, we take into account the sequence of connection flow elements instead of the repetition of edges and we match connection flows in zones to infer the similarity instead of using TF-IDF for the OGM-segment matching.

3 Activities Neighborhood Context Matching

This section elaborates our proposal to recommend activities for a business process. To achieve recommendations, we firstly present activities' contexts using graph theory (section 3.1). Secondly, we compute the similarities between activity neighborhood contexts (section 3.2). Finally, for a chosen activity, we recommend a list of activities and their involved neighborhood contexts based on the computed similarity values (section 3.3). To demonstrate our approach, we assume that there exists a 'flight-reservation' process (Fig. 4) and we are going to compute the similarity between the "Search trains" (Fig. 1) and "Search flights" (Fig. 4) activities based on their neighborhood contexts.

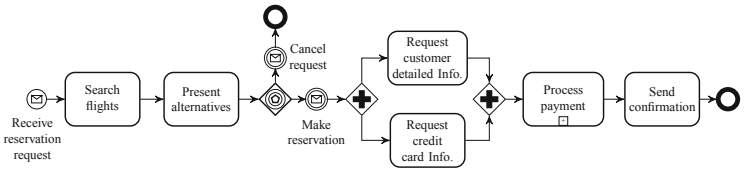


Fig. 4. Flight & hotel reservation processes

3.1 Graph-Based Activity Neighborhood Context

We choose graph theory to present a business process and an activity neighborhood context because the structure of a business process can be mapped to a graph. Without loss of generality, we select and use BPMN in our approach as it is one of the most popular business process modeling language. In our work, we define an activity or a start event or an end event as a vertex, and the sequence of *connection elements* (gateways, messages, transitions, events) that connect two vertexes as an edge (or a *connection flow*).

Definition 1 (k^{th}-layer neighbor). *A k^{th}-layer neighbor of an activity a_x is an activity connected from/to a_x via k connection flows ($k \geq 0$). The set of k^{th}-layer neighbors of an activity a_x in a business process $\mathcal{P}$ is denoted by $N^k_{\mathcal{P}}(a_x)$. $N^0_{\mathcal{P}}(a_x) = \{a_x\}$;*

Definition 2 (k^{th}-zone flow). *A k^{th}-zone flow of an activity $a_x \in \mathcal{P}$ is a connection flow which connects an activity in $N^{k-1}_{\mathcal{P}}(a_x)$ and an activity in $N^k_{\mathcal{P}}(a_x)$. Set of all k^{th}-zone flows of an activity $a_x \in \mathcal{P}$ is denoted by $Z^k_{\mathcal{P}}(a_x)$. $Z^0_{\mathcal{P}}(a_x) = \emptyset$ and $|Z^k_{\mathcal{P}}(a_x)|$ is the number of connection flows in the k^{th} connection zone of a_x.*

A path in a business process graph is called as a *connection path*. A connection path from a_i to a_j in a business process $\mathcal{P}$ is *indirected* and denoted by $CP_{\mathcal{P}}(a_i, a_j)$. The *length* of a connection path $CP_{\mathcal{P}}(a_i, a_j)$ is denoted by $Len(CP_{\mathcal{P}}(a_i, a_j))$ and the *shortest connection path* between a_i and a_j is denoted by $SP_{\mathcal{P}}(a_i, a_j)$.

Definition 3 (Activity neighborhood context graph). *Let $V_{\mathcal{P}}$ is the set of vertexes, $L_{\mathcal{P}}$ is the set of connection element names, and $E_{\mathcal{P}} \subseteq V_{\mathcal{P}} \times V_{\mathcal{P}} \times L_{\mathcal{P}}$ is the set of edges (connection flows) in the process $\mathcal{P}$. An edge $e =< a_x, a_y, P_{\mathcal{P}}(a_x, a_y) >\in E_{\mathcal{P}}$ is considered to be directed from a_x to a_y. $P_{\mathcal{P}}(a_x, a_y)$ is the string of the connection flow from a_x to a_y in $\mathcal{P}$.*

The neighborhood context graph of an activity $a_x \in \mathcal{P}$ is a labeled directed graph $G_{\mathcal{P}}(a_x) = (V_{\mathcal{P}}(a_x), L_{\mathcal{P}}(a_x), E_{\mathcal{P}}(a_x))$ where:

1. $V_{\mathcal{P}}(a_x) = V_{\mathcal{P}}$
2. $L_{\mathcal{P}}(a_x) = L_{\mathcal{P}}$
3. $E_{\mathcal{P}}(a_x) \subseteq E_{\mathcal{P}} \times N$,
   $E_{\mathcal{P}}(a_x) = \{e^x_t, e^x_t = (e_t, z_t(a_x)) : e_t =< a_i, a_j, P_{\mathcal{P}}(a_i, a_j) >\in E_{\mathcal{P}}, z_t(a_x) = Min(Len(SP_{\mathcal{P}}(a_i, a_x)), Len(SP_{\mathcal{P}}(a_j, a_x))) + 1, a_i, a_j \in V_{\mathcal{P}}\}$

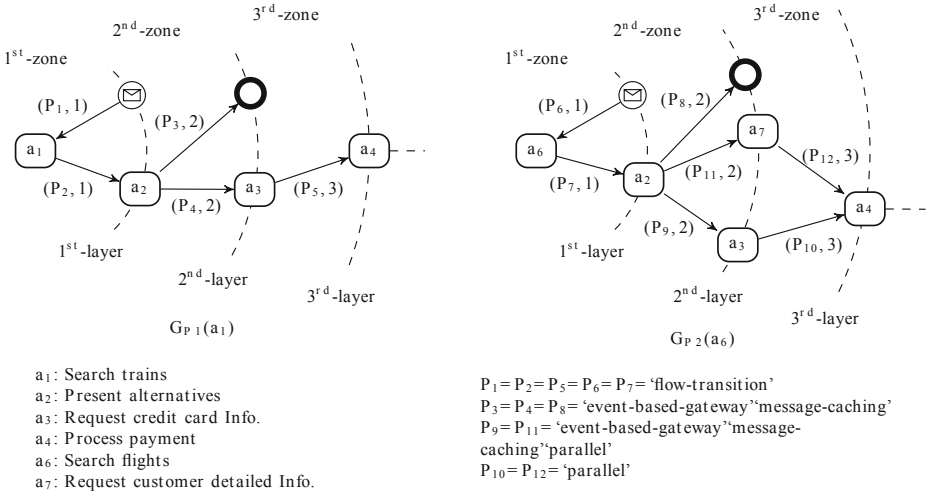


Fig. 5. Example: activity neighborhood context graph

For example, an excerpt of the "Search trains" neighborhood context graph created from "train-reservation" process (Fig. 1) and an excerpt of the "Search flights" neighborhood context graph created from "flight-reservation" process (Fig. 4) are represented in Fig. 5.

3.2 Neighborhood Context Matching

In our work, we aim at *exploiting the relation between activities* to find activities that have similar neighborhood contexts with the context of a selected activity. We propose to *match all connection flows that belong to the same connection zone and have the similar ending activities.*

Connection Flow Matching. To compute the similarity between activity neighborhood contexts, we propose to match all the connection flows connect them to/from their neighbors. Since each connection flow is a sequence of connection elements which can easily be mapped to a sequence of characters, we propose to use the Levenshtein distance [10] to compute the matching between two connection flows. Concretely, given two connection flows $P(a_i, a_j) = p_1 p_2 \ldots p_n$ and $P'(a_{i'}, a_{j'}) = p'_1 p'_2 \ldots p'_m$, their pattern matching is given by Eq. (1).

$$M_p(P, P') = 1 - \frac{LevenshteinDistance(P, P')}{Max(n, m)} \tag{1}$$

In our example, $M_p(P_1, P_6) = M_p$('flow-transition', 'flow-transition') =1; $M_p(P_4, P_9) = M_p$('event-based-gateway''message-caching', 'event-based-gateway' 'message-caching''parallel') = 0.67 and so on.

Activity Neighborhood Context Matching. The neighborhood context matching between two activities is synthesized from the matchings of associated connection flows. Besides, the behavior of an activity is stronger reflected by the connection flows to its closer neighbors. Therefore, we propose to assign a weight (w_k) for each k^{th} connection zone, so called *zone-weight* and inject this weight into the similarity computation: $w_z = \frac{k+1-z}{k}$, where z is the zone number ($1 \leq z \leq k$) and k is the number of considered zones around the activity.

Consequently, suppose that $e = (< a_x, a_y, P_{\mathcal{P}m}(a_x, a_y) >, z)$ is the edge connecting a_x and a_y by the connection flow $P_{\mathcal{P}m}(a_x, a_y)$ belongs to zone z in the activity neighborhood context graph $G_{\mathcal{P}m}(a_i)$, $e \in V_{\mathcal{P}m}(a_i)$. Similarly, $e' = (< a_{x'}, a_{y'}, P_{\mathcal{P}n}(a_{x'}, a_{y'}) >, z') \in V_{\mathcal{P}n}(a_j)$. The activity neighborhood context matching of a_i and a_j within k connection zones with the direction consideration is given by Eq. 2.

$$\mathcal{M}^k_{\mathcal{P}m,\mathcal{P}n}(a_i, a_j) = \frac{2}{k+1} \times \sum_{z=1}^{k} \frac{\sum_{e.z=e'.z'=z} \frac{k+1-z}{k} \times M^*(e, e')}{|Z^z_{\mathcal{P}m}(a_i)| - |Z^{z-1}_{\mathcal{P}m}(a_i)|} \tag{2}$$

where:

- $M^*(e, e') = M_p(P_{\mathcal{P}m}(a_x, a_y), P_{\mathcal{P}n}(a_{x'}, a_{y'}))$ if :
  - ① $(z = z' = 1) \wedge ((a_x = a_i \wedge a_{x'} = a_j \wedge a_y = a_{y'}) \vee (a_x = a_{x'} \wedge a_y = a_i \wedge a_{y'} = a_j))$
  - ② $(1 < z = z' \leq k) \wedge (a_x = a_{x'}) \wedge (a_y = a_{y'}))$
- $M^*(e, e') = 0$ in other cases.
- $|Z^z_{\mathcal{P}m}(a_i)| - |Z^{z-1}_{\mathcal{P}m}(a_i)|$ is the number of connection flows in the z^{th} connection zone of $G_{\mathcal{P}m}(a_i)$ (see Definition 2).

Return to the illustrated example, neighborhood context matching computed within three zones[1] between a_1 and a_6 (Fig. 5) is: $\mathcal{M}^1_{\mathcal{P}1,\mathcal{P}2}(a_1, a_6) = \frac{2}{3+1} \times (\frac{\frac{3}{3} \times M_p(P_1, P_6) + \frac{3}{3} \times M_p(P_2, P7)}{2} + \frac{\frac{2}{3} \times M_p(P_3, P_8) + \frac{2}{3} \times M_p(P_4, P_9)}{2} + \frac{\frac{1}{3} \times M_p(P_5, P_{10})}{1}) = 0.78$.

3.3 Activity Recommendation

The activity neighborhood context graph presents the interactions between the associated activity and its neighbors in layers. It infers the associated activity's behavior. Therefore, the matching between their neighborhood context graphs exposes the similarity between associated activities in terms of their behaviors. In our approach, the higher the matching values are, the more similar the corresponding neighborhood contexts are. For each activity in a business process, we compute its neighborhood context graph matching with others. Then, we sort the computed matching values in descending order and pick up top-N activities which have the highest matching values for the recommendation. For instance, the recommendations for the selected activities are shown in Fig. 2.

[1] The zone number can be tuned by the process designer, the more details he wants, the greater zone number is.

4 Experiments

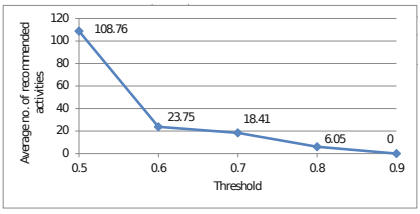
In our experiments, we aim at assessing the number of activities that have similar neighborhood contexts retrieved from a large collection of real business processes. Our goal is to two fold: (i) to show that we can find similar activity neighborhood contexts based on our proposed matching solution to prove that our approach is feasible in real use-cases and (ii) to analyze the parameters that impact the context matching computation and show the usefulness of our approach. Details of the dataset and experiments are given as following.

4.1 Dataset

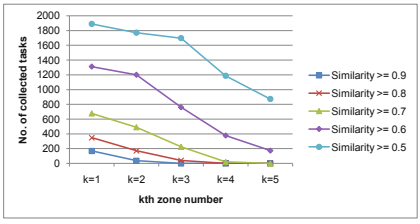
The dataset used for validating our approach is a shared collection of business process models which has been used for the experiments reported in [11]. In statistics, the collected dataset consists of 850 BPMN processes with 2203 start events, 2727 end events, 4466 activities (including 406 subprocesses), 13498 gates and 33885 sequence flows. On average, there are 8.2 activities, 2.6 start events, 3.21 end events, 39.87 interactions per process, and 5.24 gates per one connection flow. Among 4466 activities, there are 1561 activities' names existing in more than one BPMN process.

4.2 Experiments

In the first case, we set k^{th}-zone = 1 and match the activity neighborhood context graphs of all activities in the repository using the proposed computation. In results, 4373/4466 activities in the repository (97.92%) have matching values with others greater than 0, in which 1889 activities (43.20%) have matching values greater than 0.5 and 168 activities (3.84%) have matching values belonging to [0.9,1.0].



(a) Average number of recommended activities with different thresholds



(b) Number of activities having similarity >= 0.5 within 5 zones

Fig. 6. Experiments on activities recommendation

In another experiment we compute, for each activity, within three zones the average number of recommended activities that have similarity values greater than a given threshold. With 0.8 as threshold, for each activity, our approach recommends on average 6.05 activities that have similar neighborhood contexts.

We can notice that this average number of recommended activities decreases when the threshold increases as showing Fig. 6a. We noticed also the same behavior if we fix the threshold and tune the the zone number, i.e. the average number of recommended activities decreases when the zone number increases.

In the second case, we increase the k^{th}-zone value to extend our evaluation to the further layers. We get experiments with $k = \overline{1..5}$. We retrieved 4446 activities in the second zone, 4372 activities in the third zone, 4254 activities in the forth zone and 4072 activities in the fifth zone that have similarity values greater than 0. Fig. 6b shows only cropped data with the accumulated numbers of activities having similarity values greater than 0.5. These numbers decrease when k increases because our algorithm matches only the connection flows connecting two similar activities in the greater zone numbers. When k increases, the number of unmatched neighbors generally increases faster than the matched neighbors. This yields the number of unmatched connection flows increases fast and causes the reduction of similarity values in further zones.

In general, the experiments show that our approach is feasible in retrieving activities that have the similar neighborhood context in real use-cases. Based on the computed similarity, business process recommendation strategies can be run to assist the business process designer to facilitate his (re)design.

5 Conclusion

In this paper, we propose an original approach that captures the activity neighborhood context to assist business process designers with recommendations. Based on the recommended activities, the designer can easily improve or expand the process to achieve more business goals. In addition, our solution can help to create more business process variants.

In our future work, we intend to investigate the co-existence of connection flows in business processes, as well as the number of time that an activity is used in oder to refine our matching algorithm. We also aim at extending our approach to use event logs to infer the business processes for the approach's input.

References

1. Yan, Z., Dijkman, R., Grefen, P.: Fast Business Process Similarity Search with Feature-Based Similarity Estimation. In: Meersman, R., Dillon, T.S., Herrero, P. (eds.) OTM 2010, Part I. LNCS, vol. 6426, pp. 60–77. Springer, Heidelberg (2010)
2. van der Aalst, W.M.P., Alves de Medeiros, A.K., Weijters, A.J.M.M.: Process Equivalence: Comparing Two Process Models Based on Observed Behavior. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) BPM 2006. LNCS, vol. 4102, pp. 129–144. Springer, Heidelberg (2006)
3. Li, C., Reichert, M., Wombacher, A.: On Measuring Process Model Similarity Based on High-Level Change Operations. In: Li, Q., Spaccapietra, S., Yu, E., Olivé, A. (eds.) ER 2008. LNCS, vol. 5231, pp. 248–264. Springer, Heidelberg (2008)

4. Dijkman, R., Dumas, M., García-Bañuelos, L.: Graph Matching Algorithms for Business Process Model Similarity Search. In: Dayal, U., Eder, J., Koehler, J., Reijers, H.A. (eds.) BPM 2009. LNCS, vol. 5701, pp. 48–63. Springer, Heidelberg (2009)
5. Ehrig, M., Koschmider, A., Oberweis, A.: Measuring similarity between semantic business process models. In: APCCM 2007, pp. 71–80 (2007)
6. Dijkman, R., Dumas, M., van Dongen, B., Käärik, R., Mendling, J.: Similarity of business process models: Metrics and evaluation. Inf. Syst. 36(2), 498–516 (2011)
7. Weidlich, M., Dijkman, R., Mendling, J.: The ICoP Framework: Identification of Correspondences between Process Models. In: Pernici, B. (ed.) CAiSE 2010. LNCS, vol. 6051, pp. 483–498. Springer, Heidelberg (2010)
8. Sakr, S., Pascalau, E., Awad, A., Weske, M.: Partial process models to manage business process variants. IJBPIM 6(2), 20 (2011)
9. Lincoln, M., Gal, A.: Searching Business Process Repositories Using Operational Similarity. In: Meersman, R., Dillon, T., Herrero, P., Kumar, A., Reichert, M., Qing, L., Ooi, B.-C., Damiani, E., Schmidt, D.C., White, J., Hauswirth, M., Hitzler, P., Mohania, M. (eds.) OTM 2011, Part I. LNCS, vol. 7044, pp. 2–19. Springer, Heidelberg (2011)
10. Levenshtein, V.I.: Binary Codes Capable of Correcting Deletions, Insertions and Reversals. Soviet Physics Doklady 10, 707 (1966)
11. Fahland, D., Favre, C., Jobstmann, B., Koehler, J., Lohmann, N., Völzer, H., Wolf, K.: Instantaneous Soundness Checking of Industrial Business Process Models. In: Dayal, U., Eder, J., Koehler, J., Reijers, H.A. (eds.) BPM 2009. LNCS, vol. 5701, pp. 278–293. Springer, Heidelberg (2009)