Processes Summarization

Biplav Srivastava

IBM Research - India
New Delhi, India
sbiplav@in.ibm.com

Abstract

A large number of processes, of different characteristics, are being created, stored and reused during Services-Oriented Architecture based implementations. However, a user today can get little insight from such a collection of processes other than to search it with a keyword based query interface. In this paper, we introduce the problem of automatically summarizing a collection of processes and propose a comprehensive solution for it. We employ the solution on diverse collections containing hundreds of processes and demonstrate that the technique can shed correct and valuable information where none existed before, while being scalable and general-purpose.

1 Introduction

A process refers to the description of a set of coordinated activities[1]. They may come in many types during a Services-Oriented Architecture based implementation. The best known type of process is business processes which specify functionality for a to-be built system. Business process come in different notations like the graphical Business Object Model (BOM) in WBI Modeler[2] tool and Business Process Modeling Notation (BPMN[3]); or a semi-structured format like SAP's business processes in Solution Composer tool[4] (SC) represented in proprietary XML or in an unstructured format like Word files following a template, e.g., Process Definition Documents (PDDs) in SAP projects.

Another type of process is service compositions/ plans automatically created by AI planning techniques that approaches[16, 4]. The plans describe expectation from the constituent set of activities (services) that make up the process, when they will be executed in the future. Yet another type of process describes executable behavior. Workflows represented in Business Process Execution Language for Web Services (BPEL for short) and Web processes created by Co-Scripter tool[8] are instances of such processes. Table 1 is a summary of the different manifestations of processes and their salient aspects.

Now, there are many scenarios where a large number of processes are available with or without the knowledge of their provenance. As examples, vendors like SAP ship business process content with their products or after a software project is completed, its usecases representing important business processes could be accumulated as assets to be reused in other projects; an organization can store workflows and testcases of transactions done in the past; when a planner is used to generate compositions (plans) in a domain over time, the plans can be accumulated. The current approach to work with these processes is to store them in a repository, like a file system, database or commercial asset repository[5], and provide basic query support (e.g., browsing or key-words) to retrieve them. But a user will get little insight from such a repository about the stored processes by browsing or plain statistics, especially when the repository is large and she has little experience with the domain from which the processes came. In response, inspired by the automated text summarization[5] problem, we introduce the *processes summarization* problem as following.

Problem Statement: Summarize a collection of processes to reveal insights on their content without human intervention.

The challenge in solving this problem is to determine what constitutes a good summary and then building a general method which can deliver it handling the diversity of process representations. Summary depends on the eye of the user. In the established area of text summarization[5], summaries are *indicative* or *informative*. In the context of processes, if the user has no purpose in mind, we consider her summary needs to be indicative. This is indeed provided by most repositories which report on what is stored in them.

International Conference on Management of Data
COMAD 2010, Nagpur, India, December 8–10, 2010

[1] We use terms steps and actions synonymously with activities.

[2] http://www.ibm.com/software/integration/wbimodeler/

[3] http://www.bpmn.org/

[4] http://www.sap.com/solutions/businessmaps/composer/index.epx

[5] E.g., see http://www-01.ibm.com/software/awdtools/ram/

Process Type	Description About	Usage Domain	Process Content	Used in Expts?
SAP Business Processes	Common Functionality (Specification)	Business Transformation	Steps, Description, Product, Annotations	Yes
BPMN Models	Intended Behavior (Specification)	Business Modeling	Steps, Gateways, Collaborations, Events Annotations	Yes
WBI Models	Intended Behavior (Specification)	Business Modeling	Steps, Roles, Resources, Organization, Business Metrics Data, Annotations	No
PDDL Plans	Expected Behavior	Composition/ AI Planning	Steps, Annotations(Technique, Time, MakeSpan, etc.)	Yes
Web Processes	Executable behavior	Co-Scripter web scripts	Steps	Yes
BPEL4WS Workflows	Executable behavior	Execution	Steps, Messages	No

Table 1: Different manifestations of processes and their salient aspects. The last column mentions whether they were used in experiments presented.

However, most users are objective-driven when looking at process repositories and they want an informative approach which works on the content of the processes. We call such users *purpose driven*. Some purposes are: (1) Find high-level concepts in the collection (2) Find novel processes that share attributes with others and those that do not (3) Help resolve noise in collection (4) Find insights that matter to user.

Our solution, implemented in Java and called *ProcSumm*, is a flexible approach agnostic to input representation and works on any available type of process content ranging from metadata, syntactic step information, process features interpreted as semantic annotations to multi-dimensional textual content. The approach does not require all process attributes to be present and consequently, it is easy to use with simple processes (e.g., plans) while sophisticated enough for complex process representations (e.g., business processes). The summary provides insights about what the repository contains at the aggregate level as well as in subsets (clusters) of processes, built using an extensible set of process distance measures. Figure 1 is a sample of the summary output by *ProcSumm*. We employ the solution on diverse processes repositories consisting of hundreds of processes and demonstrate that the technique can shed correct and valuable information where none existed before.

Our contributions are that we: (a) formalize the problem of summarizing processes in a repository (b) present a comprehensive approach to solve the problem (b) demonstrate that the solution works on a wide variety of process representations (d) show that the technique can provide novel insights. In the rest of the paper, we give some motivations and then describe our approach, implementation and initial results. We end the paper with a detailed comparison of our approach with literature and pointers to future work. Further details are available in the longer version of the paper[14].

2 Background and Motivation

In this section, we describe three types of processes considered in the paper and the mechanisms available to work with them, leading to the need for process summarization on common process types.

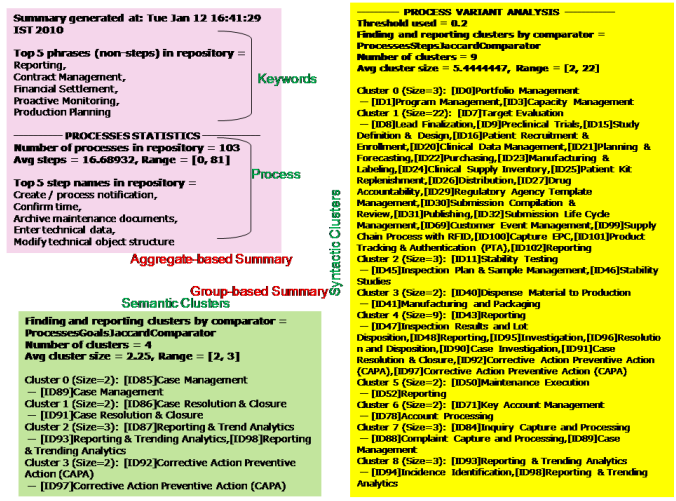

Figure 1: Summary of 103 *Pharmaceutical* processes in SAP's Solution Composer tool

2.1 Collections of Business Processes

SAP's Solution Composer tool (ver. 2.11.14) lists 620 processes prevalent in 26 industries specifying common business functionalities, which can then be implemented using its packaged middleware and third party products – databases, workflow systems and user interfaces. The XML files containing the information about the processes can be considered as repositories. Any one of them contains hundreds of processes and it is not possible to get their summary today.

We also considered processes in the increasingly popular BPMN notation. Here, we obtained 32 processes from a separate research team that had created it from different off-the-shelf tools over a period of time. Note that these BPMN processes are not straightforward translation of pre-existing processes from a different notation; hence, they document how that team has used BPMN tools.

2.2 Collections of Web Processes

Consider the CoScripter tool[8] which records web-based processes in human-readable scripts that can be shared on wikis. Figure 2 shows a simple process (se-

quential script) recorded with CoScripter to explain how to find about Mahatma Gandhi on the web using Google search and Wikipedia[6]. The script can be replayed automatically by anyone having the CoScripter plugin using a browser.

```
1. * go to "http://www.google.com"
2. * enter "mahatma gandhi" into the "Google Search" textbox
3. * click the "Google Search" button
4. * pause 20 seconds
5. * click the "Mohandas Karamchand Gandhi - Wikipedia, the
       free encyclopedia" link
6. * pause 30 seconds
7. * click the "Gandhi (disambiguation)" link"
```

Figure 2: A web process to find about *Gandhi* on the web.

More and more software created today are web-based and a web process recording tool can be used to test them. Consider a scenario where a web software project has to accomplish three functionalities: F^*, the base functionality; F_1, which reuses F^* for situation 1 and has additional functionalities ($F_1 \supseteq F^*$); and F_2, which reuses F^* for situation 2 and has additional functionalities ($F_2 \supseteq F^*$). CoScripter can be used to record how the software behaved while testing in the two situations (F_1 and F_2). Like test cases from any other software, the recorded processes can be collected and later projects will benefit from their process summarization. Within IBM, CA[10] is a web 2.0 based tool for delivering content to consultants involved in packaged application projects. There is a small but growing repository of $\tilde{2}5$ testcases where F^* is core CA, F_1 is CA for Oracle and F_2 is CA for SAP. The author could access these web processes.

2.3 Multiple Plans

With the increased focus on automated composition of services, AI planning methods[16] have become relevant to SOA. A recent paper[4] reports how SAP modules can be composed with the prominent PDDL standards. The plans describe expectation from the constituent set of activities (services) that make up the process, when they will be executed in the future. We choose plans as an example of processes.

In planning, International Planning Competitions evaluate planners on a variety of domains and make the results publicly available along with a report of the competition. One can download the plans also into their file system and search or browse the plans, but there is no easy way today for someone to gain insight about the competition or the participating planners from the data. We use the plans from IPC-5 held in 2006 [7]. Figure 3 is an example of an automatically created plan which specifies the action to take, the time at which to do it and the resources needed to take the action.

[6]The process has 7 steps but 2 are for waiting long enough for the page to get rendered despite network latency.

[7]http://zeus.ing.unibs.it/ipc-5/

```
; Time 0.00
; ParsingTime 0.00
; NrActions
; MakeSpan
; MetricValue 27.040
; PlanningTechnique Modified-FF(enforced hill-climbing
                    search) as the subplanner

0.001: (CHOOSE P300) [0.000]
0.004: (CHOOSE CDK46P3-CYCD) [0.000]
0.007: (CHOOSE CDK46P3-CYCDP1) [0.000]
0.010: (CHOOSE PCAF) [0.000]
0.013: (INITIALIZE PCAF) [0.000]
0.016: (INITIALIZE PCAF) [0.000]
0.019: (INITIALIZE PCAF) [0.000]
0.022: (INITIALIZE P300) [0.000]
0.025: (ASSOCIATE PCAF P300 PCAF-P300) [1.000]
0.028: (INITIALIZE PCAF) [0.000]
0.031: (INITIALIZE PCAF) [0.000]
0.034: (INITIALIZE PCAF) [0.000]
0.037: (INITIALIZE P300) [0.000]
1.040: (ASSOCIATE PCAF P300 PCAF-P300) [1.000]
```

Figure 3: Example of a metric plan in PDDL in Pathways domain from experiments.

2.4 Discussion

As we see, having collections of processes is becoming prevalent. A user can do a few things with them: (a) find aggregate statistics like number of processes in the repository; (b) query by keywords: this is commonly available but one has to know the terms on which to search upfront; (c) query by facets: they are pre-defined metadata to view the content of a carefully populated repository. However, the processes have to be categorized properly with the facets and moreover, process content cannot be dynamically used in search. Work on business process queries is also in this direction[11, 1, 9]; (d) query on XML: if XML is the native representation, one can use a query technique like XQuery on storage structure to find matches and differences. However, the search becomes sensitive to low-level syntax that may have nothing to do with the process content.

Thus, there is no support to summarize the collection and that is what we tackle next.

3 Solution for Summarizing Processes

Our solution for summarizing processes is shown in Figure 4. The main steps are listed below and elaborated in subsequent sections:

1. Load processes
2. Perform a variety of analyses
3. Generate summary depending on significance sought
4. Output results in desired form

We observe that there are two competing goals for *ProcSumm* design – be generic and yet have a reasonable summarization behavior without human intervention. Hence, although its modules are configurable (e.g., parsers, analyses, distance metrics), we have default settings to reduce human intervention. Automatic tuning of these defaults is an important area for future research[17].

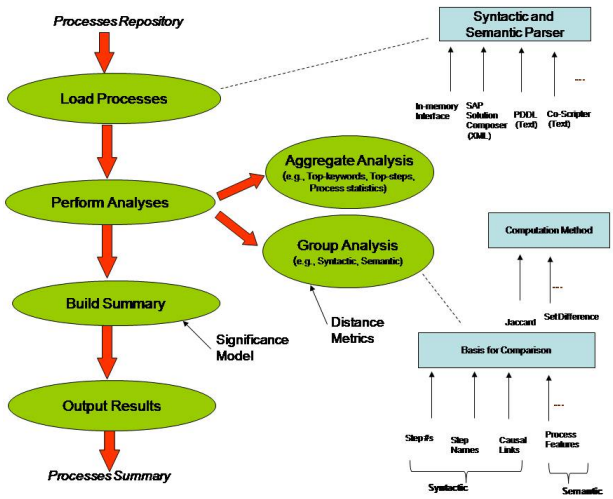

Figure 4: System Architecture of *ProcSumm*

3.1 Load processes

Although processes come in different representations from different domains, they share a common high-level semantics of representing a set of coordinated activities. *ProcSumm* can handle any process notation that can parsed into its canonical process specification consisting of: what are the activities, containing inputs, outputs, pre-conditions and post-conditions; who are the actors; what is the data that is manipulated; what are the dependencies among the activities; and what are the semantic annotations, including business policies, goals and metrics that are needed to manage the activities. For every process type (e.g., BPMN), a specific parser adapter is written according to a simple interface to read files of that type and create process datastructures made up of syntactic and semantic information therein. The choice of what constitutes each of these categories is process type dependent. In the canonical representation, syntactic features are: (a) Process Steps and Ordering, (b) Data Artifacts, (c) Business Artifacts - Resources, Roles and Organizations, and (d) Statistics; while semantic features are: (a) Goals, (b) Policies, (c) Metrics, and (d) Annotations, to account for anything else in process data. Note that this can handle all representations shown in Table 1.

3.2 Perform Analyses

Once the processes are parsed into a canonical representation, different types of analyses are possible based on the information captured. We discuss aggregate and group analyses here, and more can be easily added. In this stage of summarization, all analyses are attempted to discover insights while in the next phase, some may be filtered out. Note that depending on specific data instances, an analysis may fail if relevant data is missing.

3.2.1 Aggregate Analysis

Aggregate analysis refers to information discovered for the whole collection. The supported ones are:
(1) *Number of processes* in the collection
(2) *Top-k keywords* in the collection, for any desired k
(3) *Top-k process steps* in the collection, for any desired k

Among the above, *Top-k keywords* can be from any aspect of the multi-dimensional information in the processes of the collection. They have the potential to reveal unexpected results as also evidenced in the experiments.

3.2.2 Group Analysis

Algorithm: **FindSimilarProcesses**
Inputs: $\$D$: a set of processes, δ: the distance measure, φ: a threshold for similarity
Output: Set $\mathbf{G} = \{g_1, \ldots g_k\}$ where $g_i \subseteq \mathbf{D}$, $\|g_i\| \succ 1$, $g_i \cap g_j = \{\}$

Main Steps
1. Let $\$SP$ = *FindSimilarProcessPairs*($\mathbf{\$D}$, φ)
2. $\mathbf{G}$ = *BuildClustersbyTransitiveClosure*($\$SP$)

Sub-Routine: **FindSimilarProcessPairs**
Inputs: $\$D$, δ, φ
Output: Set **$SP** of similar process pairs
1. For each pair S_i, S_j of processes
2. Score = Calculate $\delta(S_i, S_j)$
3. Compare score with φ and decide similarity
4. Build list of pair of similar processes and return

Sub-Routine: **BuildClustersbyTransitiveClosure**
Inputs: $\$SP$: a set of similar process pairs
Output: Set **G**
1. $Clusters = {}
2. For each pair in passed $SP
3. $seed = pick first process of pair
4. $ACluster = **BuildAClusterWithGivenSeed**(i,$\$SP$)
;; Note: $\$SP$ can get reduced by side-effect.
5. Add $ACluster to $Clusters
6. Return $Clusters

Sub-Routine: **BuildAClusterWithGivenSeed**
Inputs: $seed: a process all of whose neighnors have to be found
$\$SP$: a set of similar process pairs
Output: Set g_i
1. $ClusterOfSeedProcess = {}
2. For each process i similar to seed process
3. Add i to $ClusterOfSeedProcess
4. Remove pair from $\$SP$
5. For each process j in $ClusterOfSeedProcess
6. $NewMembers = **BuildAClusterWithGivenSeed**(j,$\$SP$)
7. Add $NewMembers to $ClusterOfSeedProcess
8. Return $ClusterOfSeedProcess

Figure 5: Pseudo-code of algorithms to group processes.

The idea behind group analysis is to find subsets of processes which are common by some distance measure and this grouping can indicate a meaningful insight to the user. For grouping processes, clustering algorithms [6] provide a natural unsupervised solution framework except that the distance function has to be provided. One need not select a single measure – in our case, we try grouping with both syntactic and semantic measures on the process content. With multiple measures, one can opt to aggregate grouping results[18] or present groupings selectively based on analysis-driven significance. An issue with aggregating groupings arising from different distance measures (e.g., semantic and syntactic) is that one has to make an apriori commitment on weightage of the results. Our approach is to measure the significance in the groupings from different distance measure, including their aggregated

combination (if enabled), and let the significance drive the summary output. This allows us to minimize human intervention while retaining flexibility if one or more distance measures do indeed reveal interesting grouping patterns.

We detail our selected clustering algorithm and our novel approach to work with multiple distance measures between processes. We note that there are many alternatives for building similarity scores [3, 7] and selecting the best measure for an application is an active topic of research. Our aim is to demonstrate the overall feasibility and usefulness of process comparison approach; and the results can be further improved with better selection of distance measures.

The main steps are shown in Figure 5. The steps consist of first finding pairs of similar processes using their comparable (syntactic or semantic) contents and then using pair-wise similarity with standard transitive closure techniques to build clusters of overall equivalent processes. The rest of the section gives examples of distance measures between processes using different content structure.

Let $\delta(S_i, S_j) \rightarrow [0, 1]$ denote a distance function between a pair of processes. A value of 0 represents complete similarity of processes while 1 represents complete dis-similarity. To create a distance measure, one needs to decide the basis for comparison and the method for computation[15]. We now define some distance measures derived from syntactic and semantic content of the processes.

Syntactic Distance Measures

Steps are commonly considered as syntactic content of a process. We define two measures based on them. $\delta\#steps$ is defined on the number of steps in the processes while $\delta steps$ is defined on the number of steps common between two processes. Note that steps can repeat in a process and hence $\delta steps$ must accommodate them.

$$\delta\#steps(S_i, S_j) = \frac{\|S_i.\#steps - S_j.\#steps\|}{max(S_i.\#steps, S_j.\#steps)}$$

$$\delta steps(S_i, S_j) = 1 - \frac{\|s_i \in S_j.steps\| + \|s_j \in S_i.steps\|}{S_i.\#steps + S_j.\#steps}$$

where $s_i \in S_i$.steps, $s_j \in S_j$.steps.

Semantic Distance Measures

In the canonical process representation, annotations capture the semantic content of the process as obtained from the parser. Similar to $\delta steps$, we define a measure of semantic similarity, $\delta anns$, on the number of annotations common between two processes.

$$\delta anns(S_i, S_j) = 1 - \frac{\|s_i \in S_j.anns\| + \|s_j \in S_i.anns\|}{S_i.\#anns + S_j.\#anns}$$

where $s_i \in S_i$.anns, $s_j \in S_j$.anns.

Aggregated Distance Measures

An example of the aggregated distance measure is given with $\delta agg1$. Note that aggregated distance measures may be distracting to users when they first explore a process collection. Later, users may want to tweak the distance measure or the weighing functions by which the results are aggregated. We handle these variations easily.

$$\delta agg1(S_i, S_j) = \omega.\delta steps(S_i, S_j) + (1 - \omega).\delta anns(S_i, S_j)$$

where ω is a weighing function.

3.3 Generate Summary Based on Significance

Although a lot of analytics can be done on process content, the objective of a summary is to be concise and relevant to the user. We introduce the notion of a significance model, a set of configurable rules, whereby the user can convey an interest in differentiating fragments of process content. One such differentiation could be process' ends (e.g., goals) versus the means to achieve the ends(e.g., steps). This high to low partial order of process content can then be used to filter the analyses result in resulting summary. Another is when to report result and how to weigh aggregate measures. The significance model implemented in *ProcSumm* is:

- If size of a cluster is greater than 1, then only report that cluster in output.
- High-to-low order: When applying distance measure on process content, use the order of Goals, Annotations, Data Artifacts, Steps and then Resources.
- By default, disable aggregate measure. If enabled by user but no weightage is provided, give equal weightage to all measures.

3.4 Output Results

The aim here is to output the summary in any format the user is interested in. By default, it is in text but an XML is also produced. Using style-sheets, the output can be converted to any format for suitable consumption. Figure 1 showed a sample of an actual summary generated on a sample collection of 103 business processes in SAP's Solution Composer for *Pharmaceutical* industry.

4 Experiment

We now discuss how *ProcSumm* performs in practice. The objective of the evaluation was to see whether different types of commonly available processes can be summarized and whether the summaries are meaningful. *ProcSumm* could perform on the presented collections of hundreds of processes within seconds[8], and hence, performance is not assessed.

[8]Not more than 10 seconds per collection.

Summaries are evaluated in text summarization literature[5] by observing the compression ratio (size of summary to the original text) and retention ratio (information in summary to that in text). Summaries by *ProcSumm* can be orders of magnitude smaller than original processes when measured by size (e.g. summary of *Pharmaceutical* is 3KB while the 103 processes take 1.2MB) and was always less than 1% in experiments. For retention ratio, we present a specific case that was tested for SAP business processes. Measuring retention ratio formally is a topic of future work. The summaries in the paper are judged by an *Oracle* to check the validity of the aggregate and groupings results, and whether noise in the dataset could be detected.

4.1 Results on Business Processes

We selected 4 business processes categories from SC. The *Cross-Industry* processes reflect industry-neutral processes while 3 industry specific business processes are *Automotive*, *Pharmaceutical* and *Public Sector*. The results of summary on these processes are shown in Table 2. There were no process steps present in *Automotive* and *Public Sector* data. The results illustrate that *ProcSumm* can easily handle missing process content.

Some key observations from the summary are: (1) Even 5 top keywords in each collection gave good indicators of the domain. For example, in *Automotive*, keywords on *Vehicle Management*, *Aftersales Support*, *Extended Warehousing* and *Logistics, Sequenced Manufacturing* were returned, in addition to SAP's generic *mySAP*. (2) The maximum number of clusters are obtained with φ=0.2. As the threshold increases, the number of clusters seem to decrease while their sizes increase. (3) Clusters found using semantic features correctly identified processes in the same domain. For example, *Export Control* and *Letter of Credit* in *Cross-Industry*. (4) Clusters found using syntactic features accurately identified process variants that shared processes. We discovered that there are very few novel processes in the collection. (5) If novelty is measured by the % of processes without variants, the maximum % of novel processes varied from 1-18% with *Cross-Industry* having the least, depending on the distance measure used.

4.1.1 Towards High Retention Ratio

We tested a specific scenario to see if the summaries could retain information needed by a user to make better decisions, without requiring him to go through the original collection of processes. This exercise helps us gauge the retention ratio of summaries.

The scenario is related to traffic management and processes which can help effectively manage them. Every city has traffic. Suppose an official in a government is looking for processes related to traffic management in SAP, specifically SAP's Solution Composer (SC). The official can pose a keyword search in SC but there are many synonyms to consider - e.g., traffic, transportation, congestion, parking - and the results may not directly return process information since the keywords can match anywhere in the SC content. Another option is to browse processes but there are hundreds of them to navigate. After searching for 30 minutes, we found one under *Other Processes→ Traffic and Parking Service* which was a place holder for content to be added in future by SAP.

The option *ProcSumm* allows in this scenario is to create summaries of plausible collections and then decide based on them. Since *Cross-Industry* is relevant for all industries and *Public Sector* is relevant for governments, these are the two relevant processes collections. The 2 summaries were created in seconds and none of them had traffic management related processes, keywords, steps or semantic annotation (e.g., KPIs, product names). So, the user concluded that SC does not have processes related to traffic quickly within minutes (for us, 2).

The user could have also been conservative and created summaries for all the 26 industries in SC Ver 2.11.14. The time taken to evaluate a collection is by the user's ability to open and close the summary files. Again, the user could have come to decision within a few minutes.

In this experimental scenario, the summaries perfectly retain the information on lack of business process content in SC related to traffic management. Measuring it formally is a topic of future work.

4.1.2 Resolving Noise in Processes Collection

We next shifted focus to BPMN. We implemented a parser for BPMN2.0-compliant output of one of the tools and ran *ProcSumm* on the 32 BPMN processes. However, in the output, we detected that only 18 processes were used for the summary. On checking closely, we found that multiple variants of the BPMN formats were in the collection. Output of some of the BPMN tools were incompatible and the parser was again extended. This now covered 23 processes and the process could be iteratively expanded to cover the full collection. We note that summarization helps us discover the diversity (noise) in the collection which was supposed to be homogeneous. We also learnt from the summaries that the external team was testing the BPMN tools as most of the processes were basic control flows with dummy data objects.

4.2 Results on Web Processes

We used the 25 web processes recorded by Coscripter and mentioned in Section 2.2. They are broken into 12 for F_1, 9 for F_2 and 4 for miscellaneous. The results

Dataset	# Processes	Avg. Steps	#Syn (δ_{steps}) Clusters	#Sem (δ_{anns}) Clusters
Automotive, φ=0.2	122	0	1 {122,122,122}	13 {9,2,44}
φ=0.5			1 {122,122,122}	2 {60,2,117}
φ=0.8			1 {122,122,122}	2 {60,2,117}
Cross-Industry, φ=0.2	189	14	16 {16,2,60}	5 {36,2,166}
mySAP ERP, φ=0.5			15 {7,2,60}	2 {92,3,181}
φ=0.8			21 {6,2,60}	2 {93,3,183}
Pharma, φ=0.2	103	17	9 {5,2,22}	4 {2,2,3}
φ=0.5			8 {8,2,22}	5 {2,2,3}
φ=0.8			7 {12,3,38}	11 {3,2,4}
Public, φ=0.2	144	0	1 {144,144,144}	7 {12,2,28}
Sector, φ=0.5			1 {144,144,144}	6 {15,2,36}
φ=0.8			1 {144,144,144}	6 {16,2,37}

Table 2: Experiments on SAP Solution Composer dataset. SAP product features within process representation considered as semantic annotations. In brackets, {avg, min, max} of cluster sizes are shown.

Threshold	#Syn (δ_{steps}) Clusters	#Sem (δ_{anns}) Clusters	Comments
φ=0.2	4 {4,2,9}	2 {11,5,16}	
φ=0.5	2 {2,2}	0	Clusters with Syn
φ=0.8	0	0	None of size $\succ$ 1

Table 3: Experiments on Co-Scripter scripts. There are 25 processes with an average of 26 steps.

Problem	Planner	# Plans	Avg. Steps	#Syn ($\delta_{\#steps}$) Clusters	#Sem
PipeWorld	Downwards	23	38	2 {10, 3, 17}	1
	Satplan	16	19	3 {5, 2, 7}	1
(Metric)	SGP	30	44	3 {6, 3, 13}	1
Pathways	Downwards	30	134	2 {14, 2, 25}	1
	Satplan	9	44	3 {2, 2, 2}	1
(Metric)	SGP	30	462	3 {9, 2, 13}	1
OpenStack	Downwards	26	126	3 {7, 2, 15}	1
	SGP	30	146	5 {6, 2, 15}	1
Mixed	OpenStack-Downwards, Pipeworld-SGP	10	22	2 (δ_{steps})	1

Table 4: Experiments on plans from IPC-2006 dataset with φ=0.2. In brackets, {avg, min, max} of cluster sizes are shown.

of summary on these processes are shown in Table 3. Some key observations from the summary are: (1) The top-level keywords were not meaningful as they captured time of creating the testcases. (2) With δ_{steps}, all 9 testcases of F_2 were correctly grouped together; 8 of the 12 testcases of F_1 were subdivided into 3 further non-unitary clusters (3) Some of the processes were wrongly titled and the summary helped detect that (both measures). (4) Clusters with δ_{anns} helped detect potentially common test cases (i.e., F^*). (5) The maximum number of clusters are obtained with φ=0.2.

4.3 Results on Plans

We chose plans as a dataset because plans from previous IPCs are readily available and it is easy to observe the accuracy of summarization since the competition results are well analyzed. We selected plans from 3 planning domains created by 3 different planners competing in IPC-5 held in 2006.

The results of summary on these plans are shown in Table 4. Some key observations from the summary are: (1) The top keywords in the plans were about PDDL annotations to capture the planner and domain characteristics. (2) The plans in the clusters are in ascending order, by plan identifiers. People unaware of IPC may find it intriguing. However, if plan length is seen as a relative measure of problem complexity, it is known that problems in the competition progressively increase in hardness and correspondingly, the solution lengths rise. *ProcSumm* was able to detect this. (3) Since all plans had the same semantic annotations, they formed a single cluster when using δ_{anns}. This analysis can be automatically suppressed during summary generation right away. (4) Although other values of φ were also tried, maximum number of clusters were obtained with φ=0.2.

In order to check if the clusters could help detect and resolve noise in a plan repository, we did an experiment where plans from 2 domains by 2 different planners were mixed and then their summary was sought. This is shown in the last row, *mixed* and δ_{steps} was used. The plans from the two domains were correctly segregated.

4.4 Discussion

From the experiments, we see that *ProcSumm* can handle business processes (in SC and BPMN notations), web processes and PDDL plans; and generate meaningful summaries where none existed before. Moreover, we illustrated by example that it could retain essential information and help resolve inconsistencies in the processes collection.

5 Related Work

The closest prior work in literature comes from business processes and planning. In business process literature, [2] presents a classification of differences between processes. [3, 7] present methods to compare processes represented in graphical notation while [18] handle unstructured Word documents. They are the necessary pre-requisite techniques to build a general-purpose process summarization solution, and we leverage them. It is worthwhile to note that all previous work on distance measures seek to find a superior measure with better characteristics. However, in the context of summarization, the user does not know about

the processes apriori and hence, the notion of a single optimal measure is debatable. The paper shows that giving flexibility to users with multiple distance measures is more practical. But it is critical to find meaningful ways to work with multiple distance measures (e.g., combine many, highlight or supress selectively based on significance of data or analysis). This opens up new research avenues for future.

There has been an active line of work on methods for supporting queries over business processes[11, 1, 9]. They work by defining an abstract model of business processes and then supporting queries on the selected model. In our work, we impose a minimal model of what constitutes a process, and this is sufficient to handle its various manifestations shown in Table 1.

In planning, Myers [12, 13] has articulated the need for summarizing a plan, comparing plans and finding dissimilar plans. For this, she defined the metatheory of the domain in terms of pre-defined attributes and their possible values covering roles, features and measures. Compared to the presented work, the user manually creates features for a domain, and fills the features manually for each plan. There is no notion of the summary of the plan repository. Moreover, no significance model is used. In [15], first different measures to characterize inter-plan distances are introduced and then off-the-shelf planners are adapted to generate divergent plans. But there is no notion of summarizing a plan collection.

6 Conclusion

We introduced the problem of automatically summarizing a collection of processes and proposed a comprehensive solution, *ProcSumm*, for it. The salient features of the solution are that it works on a broad class of process representations, provides an ensemble of analyses on processes' content, allows creation of summary based on significance, and does not require human intervention but can be configured to use any inputs, if desired. We employed the solution on diverse processes collections consisting of hundreds of processes and demonstrated that the technique can shed correct and valuable information where none existed before, while being computationally efficient.

References

[1] C. Beeri, A. Eyal, S. Kamenkovich, and T. Milo. Querying business processes. In *Proc. 32nd VLDB*, pages 343–354. VLDB Endowment, 2006.

[2] R. Dijkman. A classification of differences between similar businessprocesses. In *EDOC*, 2007.

[3] B. Dongen, R. Dijkman, and J. Mendling. Measuring similarity between business process models. In *CAiSE*, pages 450–464, 2008.

[4] J. Hoffmann, I. Weber, and F. M. Kraft. SAP speaks PDDL. In *24th AAAI*, USA, July 2010.

[5] E. Hovy and D. Marcu. Tutorial on automated text summarization. In *COLING/ACL*, 1998.

[6] A. K. Jain and R. C. Dubes. Algorithms for clustering data. In *Prentice Hall Publ.*, 1998.

[7] M. E. A. Koschmider and A. Oberweis. Measuring similarity between semantic business process models. In *APCCM*, pages 71–80, 2007.

[8] G. Leshed, E. M. Haber, T. Matthews, and T. Lau. Coscripter: Automating & sharing how-to knowledge in the enterprise. In *CHI*, 2008.

[9] I. Markovic, A. C. Pereira, and N. Stojanovic. A framework for querying in business process modelling. In *Multikon. Wirtschaftsinfor.*, 2008.

[10] P. Mazzoleni-etal. Consultant assistant: a tool for collaborative requirements gathering and business process documentation. In *OOPSLA Companion*, pages 807–808, 2009.

[11] M. Momotko and K. Subieta. Business process query language - a way to make workflow processes more flexible. In *Proc. ADBIS*, 2004.

[12] K. Myers. Metatheoretic plan summarization and comparison. In *Proc. ICAPS WK. Mixed-initiative Planning and Scheduling*, 2005.

[13] K. Myers and T. J. Lee. Generating qualitatively different plans through metatheoretic biases. In *Proc. AAAI*, 1999.

[14] B. Srivastava. Summarizing processes. In *IBM Research Report RI10008. At: http://domino.watson.ibm.com/library/ CyberDig.nsf/home*, 2010.

[15] B. Srivastava, S. Kambhampati, T. Nguyen, M. Do, A. Gerevini, and I. Serina. Domain-independent approaches for finding diverse plans. In *IJCAI*, 2007.

[16] B. Srivastava and J. Koehler. Web service composition - current solutions and open problems. In *In: ICAPS 2003 Workshop on Planning for Web Services*, pages 28–35, 2003.

[17] B. Srivastava and A. Mediratta. Domain-dependent parameter selection of search-based algorithms compatible with user performance criteria. In *In AAAI*, 2005.

[18] B. Srivastava and D. Mukherjee. Organizing documented processes. In *IEEE SCC, India*, 2009.