Using Clouds for Smart City Applications

Albino Altomare*, Eugenio Cesario*, Carmela Comito*†, Fabrizio Marozzo† and Domenico Talia*†
*ICAR-CNR, cesario@icar.cnr.it
†DIMES-University of Calabria, {ccomito,fmarozzo,talia}@dimes.unical.it

***Abstract*—The increasing pervasiveness of mobile devices along with the use of technologies like GPS, Wifi networks, RFID, etc., allows for the collections of large amounts of movement data. This amount of information can be analyzed to extract descriptive and predictive models that can be profitable exploited to improve urban life. This paper presents an integrated Cloud-based framework for efficiently managing and analyzing socio-environmental data in the urban context of cities. As case study, we introduce a parallel approach for discovering patterns and rules from trajectory data. Experimental evaluation shows that the trajectory pattern mining process can take advantage from a scalable execution environment offered by a Cloud architecture.**

I. Introduction

The concept of *Smart City* has been introduced as the application of ubiquitous and pervasive computing paradigms to urban spaces focusing on developing city network infrastructures, optimizing traffic and transportation flows, lowering energy consumption, and innovative services for citizens. This is implemented by the integration of different control systems in commercial and public interest buildings, aimed at monitoring lighting and electricity, fire detection, video surveillance, access control and public address services. Smart Cities can dramatically improve the citizen's quality of life, encourage business to invest and create a sustainable urban environment.

Emerging technologies, such as wireless devices, Cloud computing and vehicular networking, promote the developments of urban computing within smart city by enabling, among other things, an alternative way for tracking and sensing exploiting people's mobile devices to track mobile events [1] [2] [3]. This leads to the generation of a large number of trajectories drawn by the users during their daily activities, that can be analyzed to discover knowledge, i.e. patterns, rules and regularities, on the user trajectories. Thus, if we have enough data to model typical behaviors, such knowledge can be used to predict and manage future movements of people [4] [5], [6].

In this paper we propose a Cloud-based architecture specifically designed for urban computing supporting smart cities. The architecture includes computing and network infrastructures integrated in a Cloud platform that interact with data source generators like sensors, smart phones and other wireless devices. The framework includes a set of services allowing to gather and collect environmental data, and to process and analyze them in order to mine social and environmental behaviors. Exploiting this information, a set of functionalities can be implemented atop the basic services allowing to improve urban planning and management [7]. In addition, an urban mobility scenario is introduced as a use case, focused on the study of trajectories followed by users to discover hidden mobility behaviors. We apply the trajectory pattern extraction methodology to a real-world dataset concerning mobility of citizens within an urban area. Experimental evaluation shows that due to complexity and large data involved in the application scenario, the trajectory pattern mining process can take advantage from a parallel execution environment as offered by the Cloud.

The paper is structured as follows. Section II gives a background on both Cloud computing and Smart City. The Cloud-based architecture is introduced in Section III. Section IV presents the trajectory analysis scenario describing the trajectory pattern detection methodology together with the designed workflow. Preliminary experimental results are reported in Section V. Section VI concludes the paper.

II. Related Work

Several Cloud enabled tools for urban planning and management have been recently proposed. Environmental Software and Services (ESS) [1] exploits the Cloud paradigm to offer a range of services for environmental planning and management, policy and decision making, world wide. Analogously, the Environmental Virtual Observatory pilot (EVOp) [2] uses Clouds to achieve similar objectives in the soil and water domains.

The European Platform for Intelligent Cities (EPIC) [8] combines the Cloud computing infrastructure with the knowledge and expertise of the Living Lab approach to deliver sustainable, user-driven web-based services for citizens and businesses.

IBM introduced Smarter City Solutions on the IBM SmartCloud Enterprise, a public Cloud platform that includes hardware, network and storage [3]. The platform provides pay-as-you-go services for urban management within cities. Those services include application software, infrastructure, networking, systems software, middleware and maintenance.

Differently from most of the systems described above, our framework has been designed to provide general-purpose services for urban planning and management within the city context. Thus, it has not been designed for a specific application domain but it can be tailored to the different aspects concerning urban management (e.g., healthcare, transportation, tourism, etc.). Moreover, the framework has been designed as a set of modular components allowing this way easy extensibility and integration of different heterogeneous components (e.g., software, data sources, etc).

III. A Cloud-based framework for Smart Cities

In this section we introduce the architecture of a framework for the implementation of services aiming at improving the planning, managing and monitoring of activities within a urban context, such as healthcare, smart transportation, smart home, smart tourism and smart public services, emergency response and public information services. The proposed architecture has been designed as a middleware substrate allowing for the integration and handling of large-scale, fragmented, cross-thematic environmental and socio-geographic data with the focus of mining human behavior from such data for urban planning and management. As such, we can envision a set of services ranging from acquiring data from disparate sources (e.g., sensors, smart phones, gps, etc) to the integration analysis and processing of such data in order to define a set of services for urban planning and management.

Due to the complexity of urban-related activities within a smart city context, we aim to provide an integrated computing environment for composing and running applications in the smart city area, leaving the user free to work at the application level and not at the middleware programming level. To this aim, one of the main objective of the framework is to assist users in formulating problems, allowing to compare different available applications (and choosing among them) to solve a given problem, or to define a new application as composition of available data and software components. The Cloud computing paradigm allows to implement the above urban-related services: facilitates data access and storage across platforms, provides on-demand computational resources, and allows for integrated processing and data analysis. The proposed architecture is based on the use of both proprietary and open source software and the integration of both private and public-available environmental databases.

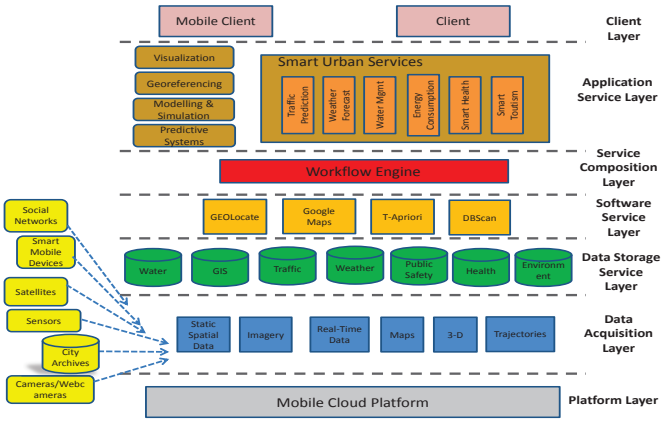

Fig. 1. A Cloud-based architecture for Smart Cities.

Figure 1 shows the architecture consisting of a set of layers. At the lower level, the *platform layer* is based on a hybrid Cloud environment (also including mobile devices at any type) that ensures cross-platform accessibility of environmental data.

The *data acquisition layer* allows to access environmental data collected from disparate sources including remote database repositories, sensor networks, mobile ad-hoc networks, satellites, camera, meters, etc. The activities found at this level can measure water quality, collect electrical meter readings for a power grid, or provide building measurements to determine its energy usage. At the *data storage level* the data collected is organized in ad-hoc repositories (historical data, recent data). The *software service layer* is composed of a set of software components (mining algorithms, GIS tools) exposed as services, that can be invoked by the upper level to compose applications. The *service composition layer* is responsible to design workflows, identify data sources, and link necessary processing components to enact the workflows. This layer is also responsible for the triggering of predefined workflows to handle the specific situations, whether it is an emergency response or the application for a citizen service. The *Smart urban application services layer* offers a set of services for urban management. As the purpose of the platform is to present a high-level management view within a set of domains, it provides means to model and define application-specific performance indicators by implementing a set of software modules each one addressing a specific urban feature. Existing applications (e.g, the outcomes from the service composition layer in application domain specific tools) can be used to perform intelligent analysis on environmental data.

The implementation of the Service Composition Layer has been done using the Data Mining Cloud Framework [9], a software environment that allow users to design and execute data analysis, mining and knowledge discovery workflows on the Cloud. Following the approach proposed in [10], such a framework models knowledge discovery workflows as graphs whose nodes represent resources (datasets, data mining tools, data mining models) and whose edges represent dependencies between resources. The framework includes a Website to compose workflows and to submit their execution to the Cloud, following a Software-as-a-Service approach.

Figure 2 shows the architecture of the Data Mining Cloud Framework, which includes a set of binary and text data containers used to store data to be mined (*Input datasets*) and the results of data mining tasks (*Data mining models*), a *Task Queue* that contains the workflow tasks to be executed, a *Task Table* and a *Tool Table* that keep information about current tasks and available tools, a pool of k *Workers* (k is the number of virtual servers available) in charge of executing the workflow tasks and finally a *Website* that allows users to submit, monitor the execution, and access the results of knowledge discovery workflows.

The following steps are performed to develop and execute a knowledge discovery application [11] (see Figure 2):

1) A user accesses the Website and develops her/his application as a workflow through an HTML-5 interface.
2) After application submission, a set of tasks that compose the workflow are created and inserted into the Task Queue.
3) Each idle Worker picks a task from the Task Queue, and starts its execution on a virtual server.
4) Each Worker gets the input dataset from its original

location.

5) After task completion, each Worker puts the result on a data storage element.
6) The Website notifies the user as soon as her/his task(s) have completed, and allows her/him to access the results.

The Data Mining Cloud Framework has been designed to be implemented on different Cloud systems. The current implementation is based on Windows Azure ("www.microsoft.com/windowsazure").

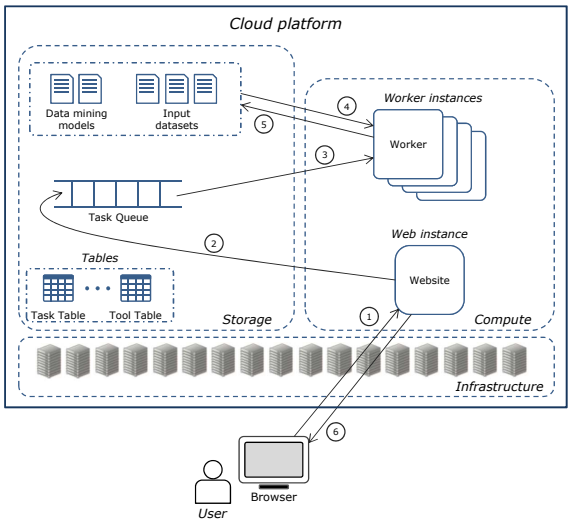

Fig. 2. Architecture of the Data Mining Cloud Framework.

IV. Trajectory Pattern Mining Application

This section presents a trajectory analysis scenario and describes a data mining approach adopted to perform a trajectory pattern detection task. Moreover, a parallel workflow implementing such a task will be introduced, that can be designed and executed in the proposed Cloud-based architecture.

Now, let us describe the approach adopted to detect trajectory patterns, whose inspiring idea is common to other approaches proposed in literature [5], [12]. Specifically, the discovery process is composed of three steps. The first step consists in the *detection of frequent regions* from the original raw trajectory dataset, whose goal is detecting spatial areas more densely passed through, in order to conduct the further analysis as movements through areas rather than single points. The second step consists in the*synthesizing of the trajectories*, by changing their representation from movements between points into movements between dense regions (i.e., each point of the original dataset is substituted by the dense regions it belongs to). The final result is the generation of the structured trajectory dataset. The third step is aimed at *extracting trajectory patterns*, in the form of associative rules, analyzing the trajectories of dense regions obtained at the previous step.

The trajectory pattern detection process consists of a sequence of concatenated connected steps, that can be parallelized, to achieve better performances in terms of execution time. Figure 3 shows a snapshot of the workflow designed through the Service Composition Layer and executed to perform a parallel trajectory pattern mining task over the Cloud. Each node represents either a data source or a data mining tool, whereas an edge represents an execution dependency among nodes. Moreover, some nodes are labeled by the array notation, which is a compact way to represent multiple instances of the same dataset or tool. For example, the *"DBSCAN[64]"*-labeled node represents 64 parallel instances of the algorithm, each one belonging to a different path of the workflow.

The workflow is composed of four steps, as described in the following:

Step 0 - Vertical Data Splitting. The original trajectory dataset, *Trajectory Data*, is partitioned in a vertical way, with respect to the timestamp value. This step is performed by the *Time Stamp Splitter* tool, which produces H data partitions. Each partition is populated by the points of the trajectories visited at the h^{th} time stamp (in Figure 3, $H = 64$).

Step 1 - Frequent Regions Detection. This step is aimed at detecting, for each timestamp, the regions that are more densely visited with respect to others (thus, of interest for the further analysis). Each $TrajPartition[i]$ dataset, $i = 1, ..., H$, is analyzed by an instance of $DBScan$ [13] and produces a $ClusteringModel$. The final result consists of H clustering models, whereas the clusters of the h^{th}-model represent the detected dense regions at the h^{th}-timestamp.

Step 2 - Trajectory Data Synthetization. This step is aimed at synthesizing the trajectories, to have a structured trajectory dataset. The *TrajectorySynthetizer* tool analyzes all models and the initial dataset, so as to generate the *Structured Trajectory Data*, where each point of the original trajectories is substituted by the dense region it belongs to. The final dataset results populated by trajectories between dense regions (but between single points).

Step 3 - Trajectory Pattern Extraction. Finally, a *Trajectory Pattern Extraction* algorithm on the dense regions trajectory data is executed, to discover trajectory patterns from them. This step has been performed by T-Apriori, an our ad-hoc modified version of the well-known Apriori algorithm [14].

V. Experimental Evaluation

We experimented the trajectory pattern methodology, described in the previous section, over the Cloud system proposed in Section III, in order to perform some experimental evaluation on a real world case. The input dataset chosen for the experiments is the *T-Drive Trajectory Data Sample* [15], [16], a collection of GPS traces describing the movement of GPS-equipped taxis in the urban area of Beijing, China. We extracted from this one a dataset composed of 80,000 trajectories, traced by 64 samples (i.e., timestamps), that has been exploited for our tests.

The experiments have been conducted using 1 virtual server to run the Data Mining Cloud Framework Website and up to 64 virtual servers for the Workers. Each test has been executed by varying the number of virtual servers used to run the trajectory pattern mining application. As performance indicators, we used the *turnaround time* and the achieved

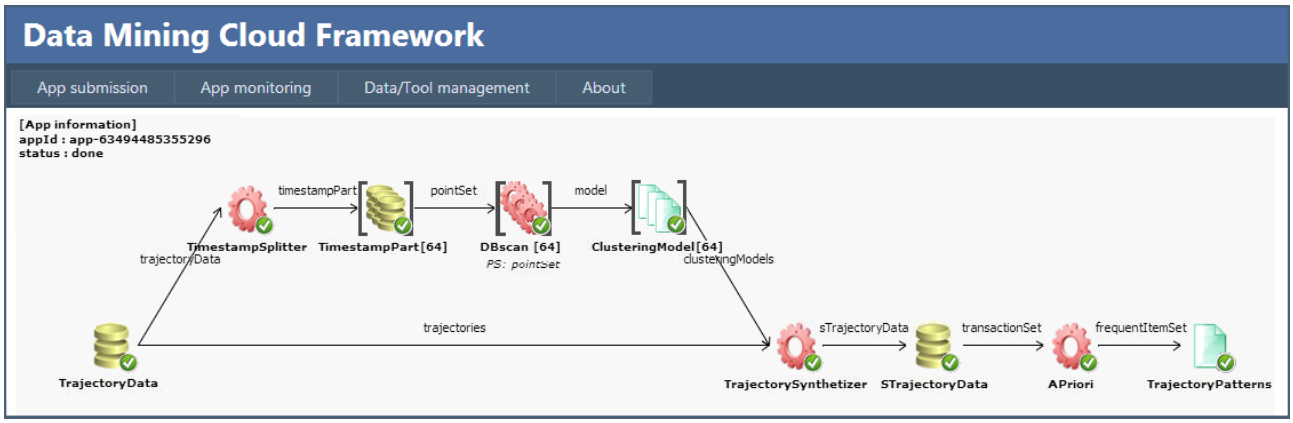

Fig. 3. Trajectories workflow at the end of the execution, with visualization of the final result.

speedup. The first one is the total time taken by the execution of the distributed algorithm, i.e., the time elapsed from the submission of the task by the client (to the system) until the final result is returned to it. The second one is the ratio of the turnaround time elapsed by exploiting 1 node to the turnaround time on n nodes.

Table I shows the turnaround times and speedups of the application using from 1 to 64 virtual servers. The case of 1 server corresponds to a sequential architecture in which the mining computation is performed on a single node. The speedup generally shows a good trend and, in particular, it is almost linear up to the case when 16 nodes are exploited.

Number of servers	Turnaround time (hh:mm:ss)	Speedup
1	34:53:53	-
2	17:33:51	1.99
4	08:50:55	3.94
8	04:30:03	7.75
16	02:20:03	14.9
32	01:14:30	28.1
64	00:41:19	50.7

TABLE I
TURNAROUND TIME AND SPEEDUP, VS THE NUMBER OF AVAILABLE SERVERS.

VI. CONCLUSION

This paper presents an architecture of a Cloud-based framework for urban computing, aimed at supporting smart cities development. Within such framework we have described a parallel approach, modeled by the workflow formalism, for pattern discovery from trajectory data. The design and execution of applications has been performed through the Data Mining Cloud Framework, a software environment allowing for execution of data analysis and mining on the Cloud.

As future work, our research will proceed to introduce some optimizations on the trajectory analysis methodology, to improve the efficiency and effectiveness of the approach. Second, we are developing a visualization module, to be integrated in the system, that can be very useful for the visualization of the input data and discovered models.

ACKNOWLEDGMENTS

This research work has been partially funded by the MIUR projects TETRIS (PON01_00451) and DICET-INMOTO (PON04a2_D).

REFERENCES

[1] "www.ess.co.at."
[2] "www.environmentalvirtualobservatory.org."
[3] "public.dhe.ibm.com/common/ssi/ecm/en/giw03021usen/giw03021usen.pdf."
[4] N. Mamoulis, H. Cao, G. Kollios, M. Hadjieleftheriou, Y. Tao, and D. W. Cheung, "Mining, indexing, and querying historical spatiotemporal data," in *Proceedings of KDD 2004*. ACM, 2004, pp. 236–245.
[5] H. Jeung, Q. Liu, H. Shen, and X. Tao Zhou, "A hybrid prediction model for moving objects," in *Proceedings of ICDE 2008*. IEEE Computer Society, 2008, pp. 70–79.
[6] E. Cesario, C. Comito, and D. Talia, "Towards a cloud-based framework for urban computing. the trajectory analysis case." in *Proceedings of CGC 2013*. IEEE, 2013, to appear.
[7] D. Talia, "Clouds for scalable big data analytics," *Computer*, vol. 46, no. 5, pp. 98–101, 2013.
[8] "ec.europa.eu/information/society/activities/livinglabs/docs/epi/cv6/pub.pdf."
[9] F. Marozzo, D. Talia, and P. Trunfio, "Using clouds for scalable knowledge discovery applications," in *Euro-Par Workshops*, 2012, pp. 220–227.
[10] E. Cesario, M. Lackovic, D. Talia, and P. Trunfio, "Programming knowledge discovery workflows in service-oriented distributed systems," *Concurrency and Computation: Practice and Experience*, vol. 25, no. 10, pp. 1482–1504, July 2013.
[11] F. Marozzo, D. Talia, and P. Trunfio, "A cloud framework for parameter sweeping data mining applications," in *Proceedings of CloudCom 2011*, 2011, pp. 367–374.
[12] A. Monreale, F. Pinelli, R. Trasarti, and F. Giannotti, "Wherenext: a location predictor on trajectory pattern mining," in *Proceedings of KDD 2009*. ACM, 2009, pp. 637–646.
[13] J. Sander, M. Ester, H.-P. Kriegel, and X. Xu, "Density-based clustering in spatial databases: The algorithm gdbscan and its applications," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 169–194, 1998.
[14] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules in large databases," in *Proceedings of VLDB 1994*. Morgan Kaufmann Publishers Inc., 1994, pp. 487–499.
[15] J. Yuan, Y. Zheng, C. Zhang, W. Xie, X. Xie, G. Sun, and Y. Huang, "T-drive: driving directions based on taxi trajectories," in *Proceedings of GIS 2010*. ACM, 2010, pp. 99–108.
[16] J. Yuan, Y. Zheng, X. Xie, and G. Sun, "Driving with knowledge from the physical world," in *Proceedings of KDD 2011*. ACM, 2011, pp. 316–324.