An architecture and algorithm for context-aware resource allocation for Digital Teaching Platforms

Fernando L. Koch[a,*], Marcos D. de Assunção[b], Carlos H. Cardonha[c], Marco A. S. Netto[c], Tiago T. Primo[a]

[a] *Samsung Research Institute, Brazil*
[b] *Inria, ENS de Lyon, France*
[c] *IBM Research, Brazil*

Abstract

Digital Teaching Platforms (DTPs) are aimed to support personalization of classroom education to help optimize the learning process. A trend for research and development exists regarding methods to analyze multimodal data, aiming to infer how students interact with delivered content and understanding student behavior, academic performance, and the way teachers react to student engagement. Existing DTPs can deliver several types of insights, some of which teachers can use to adjust learning activities in real-time. These technologies require a computing infrastructure capable of collecting and analyzing large volumes of data, and, for this, cloud computing is an ideal candidate solution. Nonetheless, preliminary field tests with DTPs demonstrate that applying fully remote services is prohibitive in scenarios with limited bandwidth and a constrained communication infrastructure. Therefore, we propose an architecture for DTPs and an algorithm to promote the adjustable balance between local and federated cloud resources. The solution works by deciding where tasks should be executed, based on resource availability and the quality of insights they may provide to teachers during learning sessions. In this work, we detail the system architecture, describe a proof-of-concept, and discuss the viability of the proposed approach for practical scenarios.

Introduction

The Digital Teaching Platform (DTP) [1, 2] is a category of solutions designed to bring interactive technology to teaching and learning in classrooms. This technology allows for the collection of multimodal data including video, audio, text, and events triggered by the interaction of students with digital content. The resulting datasets are processed to classify, understand, and predict factors that affect learning performance in relation to social and individual phenomena [3, 4]. The combination of these elements provides a continuous feed of relevant information about the level of student engagement and learning performance. Additionally, analysis of this feed enables a system to monitor meaningful signals uninterruptedly and promote adjustments towards improvement of learning performance. This scenario is the subject of research and development and is currently somewhat of a grand challenge.

An important challenge for the implementation of DTPs is to provide an adequate computing infrastructure to collect, store, and process large volumes of data, considering the requirements and characteristics of current and future digital education process [5, 6, 7, 8]. Moreover, there is a trend in using mobile devices by students and educators in teaching and learning activities [9, 10], increasing the range and scope of education and the complexity of the back-end infrastructure.

Cloud computing seems to be an ideal candidate solution to address these infrastructure challenges as it copes with the trend of distributed educational information [11, 12, 13, 14]. However, preliminary field tests demonstrated that fully remote services require communication infrastructure capabilities that currently are not available in certain locations. This is due to large amounts of data being collected and demand for: (a) low latency interactions for near real-time teacher support; (b) high quality of service to provide accurate information for decision support; and (c) optimized data transfer processes to cope with limitations on schools' communication networks and overall physical infrastructure. There is a demand for new methods of distributed computing for digital education in emerging markets that would be able to cope with limited local processing capabilities and communication networks. In addition, we

*Corresponding author: fkoch@acm.org

seek online services able to provide support to teachers while lecturing.

Motivated by this demand, we conceived an architecture for using cloud computing for Digital Teaching Platforms that, with the support of a resource allocation algorithm, supports the distribution of data analysis modules between local and remote resources by exploiting inherent features of context inference [15, 16, 17, 18]. Here, context comprises factors such as resource availability, the quality of insights teachers may demand during learning sessions and their importance, and the tasks that can be performed using locally hosted resources. The proposed model prioritizes the use of local resources to run analyses that help teachers to gauge students' engagement level during sessions. We also report results of a simulation-based evaluation of the model, showing how it maximizes the teacher's utility while coping with infrastructure resource constrains.

Challenges in DTP Data Collection and Processing

Several types of data streams are available in digital education environments. For instance, we are testing with video capturing for visual sentiment analysis with respect to the content being presented. We also experimented with environmental audio analysis to determine the level of activity in the classroom or even estimate students' stress level. Moreover, analysis of events captured from the students' interactions with digital content result in methods to infer levels of activity and attention [19].

The complete scenario is described as follows (see Figure 1 and numbers in parentheses). Sensors in the classrooms collect signals, video, and audio files, which are transmitted to local servers and (1) stored in a local repository (cache) of data and events. This information is processed by (2) low-latency algorithms tailored to estimate student education indicators, *e.g.*, level of activity and attention, which teachers can use to decide on learning activities to improve student's performance. An Analysis Coordination (AC) algorithm (3) plays the central role in the composition, supporting the distribution and configuration of local and remote data analysis modules. The remote server implements (4) advanced modules for thorough analysis of education parameters, considering aggregated data from multiple classrooms. This processing populates a (5) repository of aggregated events and a (6) repository of classified information, containing inferred profiles of students, groups, teachers, and material. An inferred profile is composed of information that was not directly provided by the user, but inferred based on data that has been collected. At this level, social learning and predictive learning analytics can be applied to the data stored in the repositories. Educators and administrators apply this information to assess learning performance and to support decision-making with respect to adjustments of the education process and material.

The first challenge detected in this scenario relates to the *volume of data*. Observations from state-of-the-art DTP technologies yield the following numbers: 60 MB to 80 MB of event log files produced per student per hour, or around 1.3 GB per hour for a classroom of 30 students, which is the Brazilian average for traditional schools. Also, there is 1.5 GB per hour of video per camera (using 720p resolution and MPEG4 compression), with two cameras normally being used. A front camera is used for face sentiment analysis and presence detection, and a rear camera is used for teacher movement analysis and class recording. Such volumes of data cannot be transmitted for remote processing even using fast network connections due to performance challenges.

The second challenge refers to *expected response time* (or *feedback velocity*). The information collected by a DTP must be processed in order to enable multimodal evaluation of students' performance and, ultimately, to support real-time recommender systems for learning activities. The response time of an algorithm typically depends on its configurations; *i.e.*, more detailed analysis conducted on more complex data typically requires more time, which generally achieves higher quality of the results.

Moreover, there is a challenge associated with the *granularity level* with which data is collected, and we may consider an *aggregated level* and *individual level*. The first category refers to data describing how students are reacting as a group. For instance, videos recorded by cameras placed in the classroom's corners, devices such as sound detectors, light sensors, and thermometers can be used to capture aggregated-level information in classrooms. The second category describes and stores how each student interacts with educational material, *e.g.*, time spent over an educational activity, screen positions touched by students during a class, written comments, and textual highlights. This data can be recorded because tablets being used by students in the classroom are equipped with sensors and logging applications that register this type of information.

Each analysis requires a *level of accuracy* depending on the purpose of the information being delivered. For instance, data accuracy may be lower in group analysis than in individual assessments. Finally, there is the

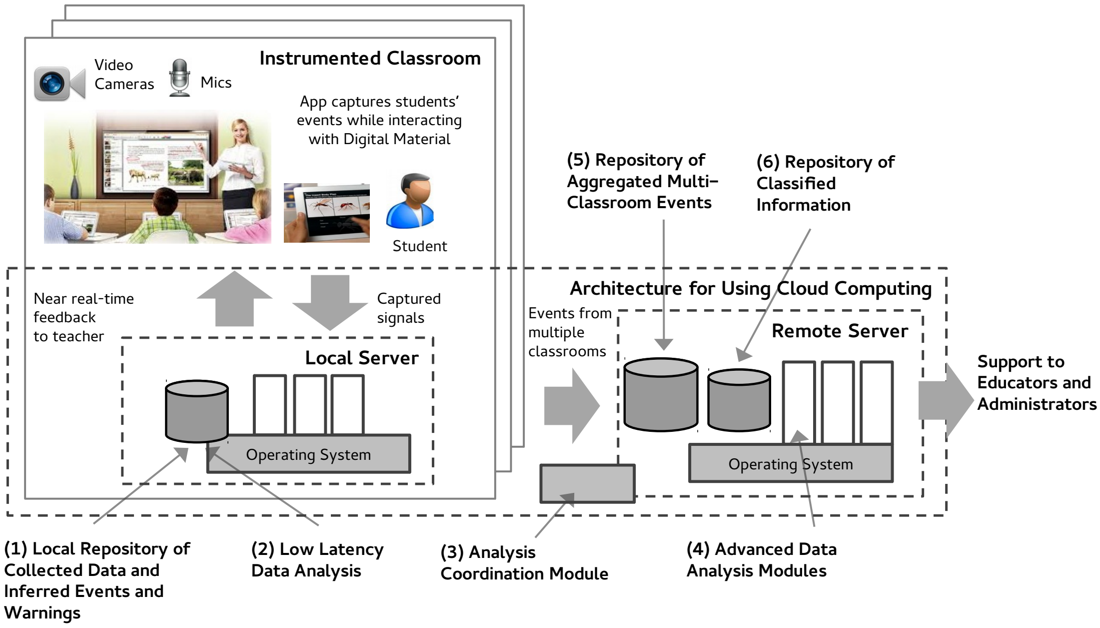

Figure 1: Architecture for using cloud computing for a Digital Teaching Platform.

problem of *processing distribution*. Some of the algorithms employed by learning analytics and social analytics are highly parallelizable, in the sense that they need very restricted (or local) input data, and their execution does not require communication with other tasks. Such tasks are typically restricted to data produced by an individual learning session. For instance, assessing differences on the performance of student(s) attending a particular classroom is a highly parallelizable task. Conversely, analysis requiring comparisons and correlations between datasets collected in several sessions depend on data aggregation.

The elements described above comprise the set of parameters affecting the execution of analytics tasks typically executed by a DTP. Examples of algorithms and insights are (i) a recommender system to support class composition by suggesting complementary educational content; (ii) a search engine to retrieve educational content; (iii) an attention-level measure that evaluates whether the students are following the educator during class; (iv) a tag cloud, based on the educational content; and (v) a post-class recommender system that analyses students' performance and recommends complementary educational content for homework activities. Choices are restricted by the amount of computational resources that the local system provides, so one would ideally choose a subset that maximizes teacher's utility, which in practice translates to information that may be employed in real-time.

In this paper, we investigate the use of insights produced in real-time for learning sessions conducted in schools equipped with low-end computers, with limited access to broadband Internet connections, and with restricted access to cloud computing solutions. In these situations, computational tasks may neither be completely delegated to remote servers nor entirely processed in the local computational environment, and teachers can only use information generated by algorithms executed locally. Therefore, there is a need for an optimal balance between local and remote processing. We must exploit inherent features of digital education scenarios to develop tailored techniques for processing distribution, such as features of the educational material (material profile), students' social setting (students profile), and pedagogical setting (teacher profile) (see Table 1).

In summary, the challenge we address in this article involves how to select algorithms to be executed in real-time (or, similarly, on the local server) given constraints related to volume of data, feedback velocity, level of accuracy, processing distribution, and resource limitations by exploring contextual information in learning scenarios.

Finally, we note that there are several ethical and pedagogical challenges involved in the deployment of digital education technologies. In practice, experiments and

Table 1: Profile parameters in digital education.

Parameter	Reason	Applicability
Material profile	Predict data complexity based on characteristics of content; more input points will lead to more log entries.	Adjust configuration to cope with more entries depending on demand for latency; *e.g.* reduce sampling rate.
Group Profile	Predict students' performance based on task resolution and interactions with content. Predicts their level of attention, orderliness, and activity.	Adjust algorithm distribution configuration to cope with the stream of students interactions data and the precision of the prediction feedback.
Teacher Profile	Predicts teachers' demand for information during classroom and activity performance.	Adjust algorithm distribution and configuration to cope with level of information demand and activity.

tests involving such platforms typically involve undergraduate students (*i.e.*, adult individuals), who are aware of all data being collected and who provide explicit authorization for the use of their data. Additionally, data anonymization is employed in order to minimize potential issues with privacy.

Using Cloud Computing in DTP

The proposed architecture relies on an analysis coordination module to distribute processing of analytics algorithms between two layers of resources: a local layer, containing computational resources located in the schools, and a remote layer, which is typically cloud-based. Additionally, it must select adequate configurations for each algorithm.

The problem consists of selecting a set of algorithms to be executed on the local layer; these algorithms, applied to data being dynamically generated by sensors installed in the classroom, generate insights and recommendations that may be employed by teachers in real-time. Each algorithm admits different configurations, which affect the volume of resources and the quality (or utility) of the results they produce. Ideally, a method will select a set of algorithms, together with their respective configurations, that will respect the local layer's processing capacity and deliver high utility (*i.e.*, provide high-quality results).

A formal description of the problem is as follows. We have a set of algorithms and, for each $a \in A$, we have a set $c(a) \in C$ of configurations. Given an algorithm a, a configuration c in $c(a)$, and an input data i, the quality of the results (or utility) is given by function $q(a, c, i)$ and the associated resource consumption by $r(a, c, i)$. Moreover, given a and i, we assume that changes on c lead to correlated changes on $q(a, c, i)$ and $r(a, c, i)$; *i.e.*, changes in c that increase $q(a, c, i)$ will also increase $r(a, c, i)$ (for more quality, one needs to spend more resources). Each dataset i is prone to variations in size and even type, but we can assume that the performance of algorithms in A depends only on the level of complexity $l(i)$ of i, and not on i specifically.

We assume that a solution's utility index is proportional to the sum of the quality of all results that were produced in the local layer. Quality in this context will ideally reflect the level of interest a teacher has on each individual result. For instance, it may be equal to the frequency with which these results have been used in previous learning sessions; alternatively, it may also reflect a rank of the algorithms provided a priori by the teacher, thus allowing for personalization from her perspective. Teachers cannot use information extracted from the remote layer during a learning session, so algorithms executed in this layer do not contribute to the solution's utility. Moreover, if $q(a, c_1, i) > q(a, c_2, i)$, then all the information produced by a with configuration c_2 is also produced by with configuration c_1, that is, one does not have any incentive to execute the same algorithm with two different configurations.

Based on the description above, our problem is to determine a set of pairs $S \subseteq A \times C$ maximizes

$$\sum_{(a,c)\in S} q(a, c, i)$$

The set of pairs is subject to a restriction on the number of available resources, given by:

$$\sum_{(a,c)\in S} r(a, c, i) \leq R$$

where $R \in \mathbb{N}$ denotes the number of computational resources provided by the local layer. One can clearly see that this is a variation of the *Knapsack Problem*, a classical NP-complete combinatorial optimization problem [20, 21]. This problem clearly belongs to the complexity class NP. Namely, the classical *Knapsack Problem* can be reduced to it by constructing instances where $|c(a)| = 1$.

Algorithm 1 contains the description of *Knapsack Scheduler*, a pseudo-polynomial algorithm that solves the problem. Notice that we use $q(a, c)$ and $r(a, c)$ instead of $q(a, c, i)$ and $r(a, c, i)$, respectively, as we do not need to represent i explicitly in Algorithm 1.

Knapsack Scheduler employs a dynamic programming approach that is very similar to the one typically

Algorithm 1: Knapsack Scheduler (A, C, R).

for $(a, r) \in A \times R$ **do**
  $S(a, r) = 0$
for $c \in c(a_1)$ **do**
  for $r(a_1, c) \leq r \leq R$ **do**
    $S(a_1, r) = max[S(a_1, r), q(a_1, c)]$
for $2 \leq i \leq |A|$ **do**
  for $c \in c(a_i)$ **do**
    for $0 \leq r < r(a_i, c)$ **do**
      $S(a_i, r) = max[S(a_i, r), S(a_{i-1}, r)]$
    for $r(a_i, c) \leq r \leq R$ **do**
      $S(a_i, r) =$
      $max\{S(a_{i-1}, r), S[a_{i-1}, r - r(a_i, c)] + q(a_i, c)\}$
return S

used to solve the *Knapsack* problem. Note that the optimal utility value is presented in $S(a_{|A|}, r)$ and that an optimal configuration can be directly extracted from S. Finally, we note that one may substitute $q(a, c, i)$ for some utility function $u(a, c, i)$ and employ *Knapsack Scheduler* as it is just by changing $q(a, c)$ for $u(a, c)$ accordingly.

Computational Experiment

In this section, we present implementation details of our approach and describe experimental results that use synthetic workloads created with information gathered during a proof-of-concept deployment.

Technical Description

We assembled a proof-of-concept experiment to evaluate the effectiveness of Knapsack Scheduler with variations of deployment parameters. The system architecture, depicted in Figure 2, comprises a local server and provides a Representational State Transfer (REST) Interface for Data Entry and Information Query (see (1) in Figure 2), including the input of data from sensors, video, and audio, along with a query of stored data, compiled events, and recommendation systems to provide support for teachers during the class.

The system also provides a Local Repository of Collected Data and Inferred Events and Warnings (see (2) in Figure 2), which concentrates raw data from sensors. Inferences are realized through this data, and the warnings are generated. An example is an inference algorithm that notifies the educator regarding students who have not interacted with the application for a long period of time.

The Configurable Services for Low Latency Data Analysis (see (3) in Figure 2) compile information required for near real-time decision support and warning. The selection of services should take into account the limitations of the platform. For instance, if the recommender system requires a certain amount of processing and network bandwidth during a class with limited hardware and network resources, this service might not be executed.

The Service for Analysis Coordination (4) controls the distribution and configuration of both local and remote data analysis modules. For instance, an inference algorithm for attention level can only be performed by one system with determined characteristics.

Inference tasks are accomplished based on information stored in Repository of Aggregated Multi-Classroom Events (5) and Repository of Classified Profile Information (6). These repositories are populated by both (a) data uploaded from local servers at the end of classes and (b) compiled data, from the processing of Configurable Services for Data Analysis Modules (7). The composition also provides a REST interface (8) for querying information, through which analytic services and coordination applications can access information from the platform, supporting, for instance, analytic reports, event information, and warnings.

Supported by the proposed architecture, the local server is deployed on a low-end computer (*e.g.*, Raspberry Pi, an Intel Next Unit of Computing), running Linux, a REST Web Application Programming Interface (API), and a Java Virtual Machine. The remote server runs on a cloud environment with a similar configuration.

The data analysis modules are programmed in Java and/or Python, facilitating the portability of modules between the platforms, so long as the required libraries are loaded on both ends. The content delivered to students during learning sessions comprises chapters that contain learning objects. Students' interactions with the content and sensors deployed on the devices used in class generate events that are initially uploaded to the local server. Each event is a tuple (event:StudentID, EducationalContentID, EventID) and requires 617 bytes of storage on average. Table 2 presents examples of inferences, recommendations, and contextualized search engines.

Experimental Setup and Metrics

In order to evaluate the performance of the proposed approach, we considered two variations of configuration parameters regarding educational settings and operational configurations. The first variation includes

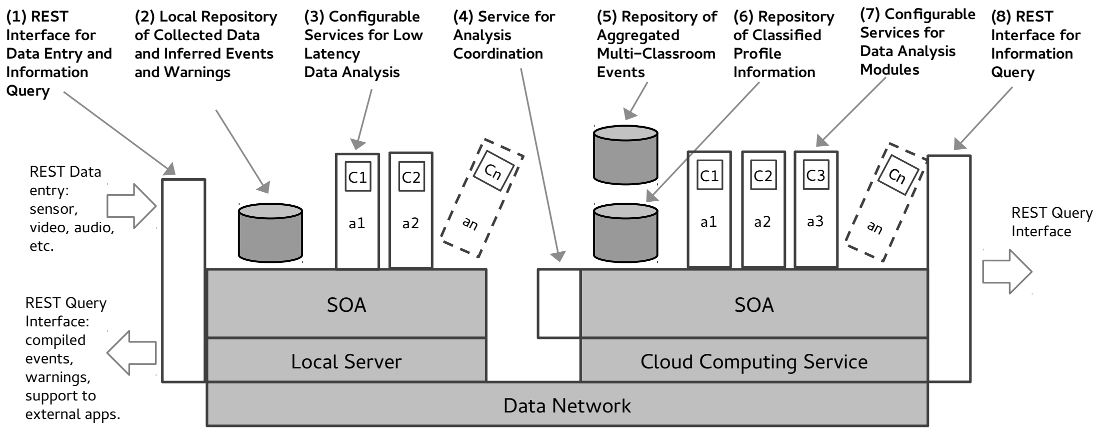

Figure 2: System architecture. (SOA: service-oriented architecture.)

Table 2: Algorithms for inference, recommendation, and search.

Name	**Description**
Activity Recognition supported by sensors	Detects and identifies the gestures from users in classroom environments, using a tablet, to infer the students' interest in the contents presented during the class.
Search of Learning Contents	Interface to search Learning Objects based on content target, contextual profile, student profile, and a natural language textual input.
Context Inference from gestures and handling of digital content	Algorithm to detect and classify gestures from users in classroom environments, using a tablet, to identify students' interest in the contents presented during the class. Currently provides a feedback to the professor about the students' attention levels.
Material Optimization Recommendation	Mathematical-probabilistic analysis of a combination with respect to content parameters, individual and group profiles, and contextual classification (from other methods) in order to find the fittest combination set of Learning Objects.
Recommendation of Complementary Material	Method to build and analyze sets of students profiles, and, based on the set of Learning Objects, automatically suggest complementary Learning Objects for each set of <*students profiles, Learning Objects*>.
Group Attention Level Assessment	Infer and evaluate the level of attention in a group of students based on the analysis of Activity Recognition supported by sensors; for instance, asses the matching between content being presented by teacher and visualized by students.
Visual Sentiment Analysis	Method to analyze video for facial expressions to provide sentiment analysis inferring the individual apathy with respect to presented material.
Visual Movement Tracking Analysis	Track level of body movement among the group of students to infer the level of activity and the social behavior.
Audio Activity Analysis	Analyze fluctuations in audio level from open microphones to infer the level of activity and the social behavior.

variations on material profiles' and student profiles' and how they affect the demand for computational resource and network distribution. The second involves variations in the distribution of computational processing and how this affects the quality of service. We then analyzed how these configurations affect the overall system performance and perceived utility.

For the experiments, we simulated the synthetic workloads based on empirical observations with respect to classrooms with state-of-the-art DTP technology, composed of a set of classes C. In this scenario, each subject lasts for 60 minutes and consists of a number of chapters uniformly distributed between 5 and 7; each chapter is made available at a specific time during class and contains from 6 to 10 learning objects, which can be text, image, or video.

We observed that approximately 20% of the learning objects are text, 30% are images, and 50% are videos. A classroom is composed of 20% highly active students (producing an average of 60 events per learning object), 60% normally active students (producing an average of 28 events per learning object), and 20% low active students (producing an average of 12 events per learning object). This information is used to generate a stream of events to be processed. The analyses are performed every 15 seconds. For each class, one machine with four processor cores was simulated to carry out the analyses.

We considered scenarios involving nine algorithms (or analyses), as illustrated in Table 2. We employed three data sample sizes , namely 0.3, 0.6, and 1.0, which determine the number of events that are considered for processing, out of the total number of events that arrived in a given time frame. Moreover, each analysis is associated with some value $v \in [0, 1]$ representing its importance. Quality $q(a, c, i)$ of analysis under configuration is given by 2^{sv}, that is, if the importance v of a is 0.7 and the sample size for c is 0.6, then $q(a, c, i) = 1.34$. The time required to execute an analysis configuration on a dataset depends on the total number of events to process, the complexity of the analysis, and the sample size.

At every 10 minutes, our platform executes a selection algorithm that determines, for the next 10 minutes, the analysis configurations that should be carried out locally, and what should be offloaded to the cloud. The goal is to execute locally the analyses that maximize the overall quality, which, as mentioned earlier, translates to maximizing the teacher's utility. In order to determine the resource demand of selected analyses, the selection algorithms consider the number of events that arrived during the previous 10 minutes.

Two algorithms are considered for selecting the analyses, namely *Knapsack Scheduler* (described beforehand) and a *greedy* algorithm that orders configurations by their expected quality and selects those whose resource demands do not surpass the resource capacity of the local machine.

Two metrics were used to evaluate the selection and placement of analyses: (1) Aggregate teacher's utility, that is, the sum of qualities of configurations at each selection interval, and (2) Number of analysis tasks offloaded to the cloud, the number of analysis tasks that are offloaded to be executed in the cloud, thus incurring delays and slower response times.

We vary the number of classes from 50 to 100 to evaluate the approach under different numbers of classes and students. All presented results are averages of 5 simulation rounds using different simulation seeds for the random number generators at each round.

With analysis (1), we want to determine which algorithm performed better considering the different parameters of event processing; such processing is important to determine the quality of the results (utility). Analysis (2) determines which algorithm led to greater reductions in bandwidth and cloud processing.

Results and Analysis

Figure 3 summarizes the results on aggregate teachers' utility. As we can observe, the utility delivered by *Knapsack Scheduler* is clearly superior to the one yielded by the greedy approach. In our experiments, the allocation problems associated with learning sessions were pairwise independent, so the results suggest that *Knapsack Scheduler* is consistently better, as its superiority was increasing together with the number of learning sessions considered in the simulation.

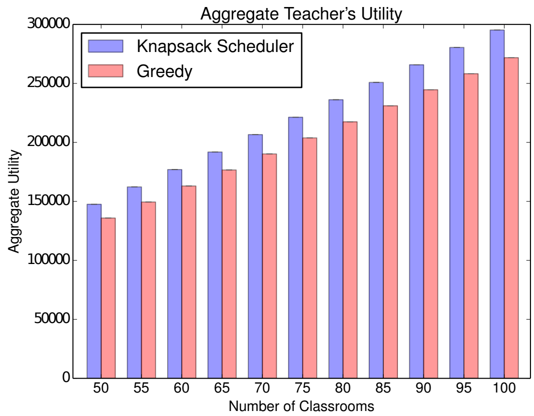

Figure 3: Aggregate teacher's utility delivered by configurations executed locally

We note that this superiority is closely related to the definition of the utility function. The values of $q(a, c, i)$ used in our experiments are such that differences between high-value and low-value utilities are small, so the greedy approach may choose analyses that are only marginally better but require much more resources.

These results cannot be considered surprising. Classic results coming in approximation theory show that it is possible to construct synthetic instances for which the greedy algorithm for the *Knapsack Problem* will deliver arbitrarily bad solutions. Note that this result cannot be directly transposed to our problem, as our algorithm does not "know" the complete dataset beforehand (in our experiments, we select the set of analysis before the arrival of student's data). Nevertheless, our results suggest that the proposed algorithm better utilizes the local resources and delivers superior utility to teachers in practical settings.

Another interesting fact about the results of our experiments related to the number of tasks that are offloaded to the cloud in real-time. The exponential utility function induced *Knapsack Scheduler* to pick all analyses in all instances with minimum sample ratio. Although individual quality was sacrificed, *Knapsack Scheduler* allowed the instructor to extract more information from the dataset. Additionally, note that such solutions avoid the transmission of data for execution in the remote layer; this aspect has not been considered in this paper, but we note that it is also of practical relevance (reductions of bandwidth consumption are also important in the scenarios we are considering).

Finally, we observe that modifications in the definition of the utility function may change the magnitude of the differences between *Knapsack Scheduler* and *Greedy*. Nevertheless, other experiments (omitted due to space limitations) also demonstrated *Knapsack Scheduler* consistently delivering better utility values (and sending less tasks for processing in the remote layer).

Conclusion

In this paper, we have introduced an architecture for using cloud computing for a Digital Teaching Platform. It promotes the intelligent coordination of the data analyses between local and remote resources by exploiting inherent features in context inference. We developed *Knapsack Scheduler*, an algorithm that balances (i) predicted distribution of intelligent algorithms and (ii) predicted configurations to be applied. The challenge is to cope with the computationally expensive process of evaluating the quality of solutions.

As simulations based on data collected from a pilot deployment demonstrate, *Knapsack Scheduler* selects the analyses that deliver the best overall utility and reduces the number of tasks that are offloaded to the cloud, hence better utilizing the locally available resources.

Acknowledgments

This work has been supported and partially funded by FINEP (Financiadora de Estudos e Projetos)/MCTI (Ministério da Ciência, Tecnologia e Inovação), under subcontract no. 03.11.0371.00. This work has also been supported by Samsung Eletronica da Amazonia Ltda as part of the project "Ecosystem of Integrated Digital Education", registered with the Brazilian Council for Scientific and Technological Developmen (CNPq) as research project 82015001.

References

[1] C. Dede, J. Richards, Digital teaching platforms: Customizing classroom learning for each student, Teachers College Press, 2012.

[2] P. Goodyear, S. Retalis, Technology-enhanced learning, Sense Publishers, 2010.

[3] P. Blikstein, Multimodal learning analytics, in: Proceedings of the third international conference on learning analytics and knowledge, ACM, 2013, pp. 102–106.

[4] K. Verbert, N. Manouselis, X. Ochoa, M. Wolpers, H. Drachsler, I. Bosnic, E. Duval, Context-aware recommender systems for learning: a survey and future challenges, Learning Technologies, IEEE Transactions on 5 (4) (2012) 318–335.

[5] C. Plesch, C. Kaendler, N. Rummel, M. Wiedmann, H. Spada, Identifying areas of tension in the field of technology-enhanced learning: results of an international delphi study, Computers & Education 65 (2013) 92–105.

[6] T. Anderson, D. Whitelock, The educational semantic web: Visioning and practicing the future of education, Journal of Interactive Media in Education 1 (2004) 1–15.

[7] D. T. Seaton, Y. Bergner, I. Chuang, P. Mitros, D. E. Pritchard, Who does what in a massive open online course?, Communications of the ACM 57 (4) (2014) 58–65.

[8] O. C. Santos, J. G. Boticario, Requirements for semantic educational recommender systems in formal e-learning scenarios, Algorithms 4 (2) (2011) 131–154.

[9] O. R. Pereira, J. J. Rodrigues, Survey and analysis of current mobile learning applications and technologies, ACM Computing Surveys (CSUR) 46 (2) (2013) 27.

[10] F. Martin, J. Ertzberger, Here and now mobile learning: An experimental study on the use of mobile technology, Computers & Education 68 (2013) 76–85.

[11] S. Dietze, H. Q. Yu, D. Giordano, E. Kaldoudi, N. Dovrolis, D. Taibi, Linked education: interlinking educational resources and the web of data, in: Proceedings of the 27th annual ACM symposium on applied computing, ACM, 2012, pp. 366–371.

[12] S.-W. Hung, M.-J. Cheng, Are you ready for knowledge sharing? an empirical study of virtual communities, Computers & Education 62 (2013) 8–17.

[13] A. Ruiz-Calleja, G. Vega-Gorgojo, J. I. Asensio-Pérez, M. L. Bote-Lorenzo, E. Gómez-Sánchez, C. Alario-Hoyos, A linked data approach for the discovery of educational ict tools in the web of data, Computers & Education 59 (3) (2012) 952–962.

[14] K. Kumar, J. Liu, Y.-H. Lu, B. Bhargava, A survey of computation offloading for mobile systems, Mobile Networks and Applications 18 (1) (2013) 129–140.

[15] M. D. de Assuncao, M. A. S. Netto, F. Koch, S. Bianchi, Context-aware job scheduling for cloud computing environments, in: IEEE/ACM Fifth International Conference on Utility and Cloud Computing (UCC 2012) Workshops, IEEE Computer Society, Washington, USA, 2012, pp. 255–262. doi:10.1109/UCC.2012.33.
URL `http://dx.doi.org/10.1109/UCC.2012.33`

[16] F. Koch, M. D. Assunçao, M. A. Netto, A cost analysis of cloud computing for education, in: Economics of Grids, Clouds, Systems, and Services, Springer, 2012, pp. 182–196.

[17] J. H. Miller, S. E. Page, Complex adaptive systems: an introduction to computational models of social life: an introduction to computational models of social life, Princeton university press, 2009.

[18] G. Siemens, P. Long, Penetrating the fog: Analytics in learning and education., EDUCAUSE review 46 (5) (2011) 30.

[19] F. Koch, C. Rao, Towards massively personal education through performance evaluation analytics, International Journal of Information and Education Technology 4 (4) (2014) 297–301.

[20] M. R. Garey, D. S. Johnson, Computers and intractability: a guide to np-completeness (1979).

[21] A. Fréville, The multidimensional 0–1 knapsack problem: An overview, European Journal of Operational Research 155 (1) (2004) 1–21.