Confidence-driven Early Object Elimination in Quality-aware Sensor Workflows *

Lina Peng K. Selçuk Candan
Computer Science and Engineering Dept.
Arizona State University
Tempe, AZ, 85287, USA.
{lina.peng,candan}@asu.edu

ABSTRACT

Distributed media rich systems, which can provide ubiquitous services to human users, require perceptive capabilities, transparently embedded in the surroundings, to continuously sense users' needs, status, and the context, filter and fuse a multitude of real-time media data, and react by adapting the environment to the user. Designing such real-time adaptivity into an open reactive system is challenging as run-time situations are partially known or unknown in the design phase and multiple, potentially conflicting, criteria have to be taken into account during the runtime. The ARIA media workflow architecture [4, 18, 19, 20], which is composed of adaptive media sensing, processing, and actuating units, processes, filters, and fuses sensory inputs and actuates responses in real-time. Unlike traditional workflows, a media processing workflow needs to capture inherent redundancy and imprecision in media, in terms of alternative ways of achieving a given goal. The object streams are only *statistically* accurate due to the inherent uncertainty of feature extractors. In this paper, we present a quality-aware early object elimination scheme to enable informed resource savings in continuous real-time media processing workflows.

Categories and Subject Descriptors

C.1.3 [**Processor Architectures**]: Other Architecture Styles—*Data-flow architectures*; C.2.4 [**Computer-communication Networks**]: Distributed Systems—*Distributed applications*; I.4.8 [**Image Processing and Computer Vision**]: Scene Analysis—*Sensor fusion*; J.5 [**Arts and Humanities**]: Performing arts; G.3 [**Probability and Statistics**]: Distribution functions

General Terms

Algorithms, performance, reliability

*This work has been supported by NSF grant # IIS-0308268, "Quality-adaptive Media-flow Architectures to Support Sensor Data Management."

DMSN'05, August 29, 2005, Trondheim, Norway.

Keywords

Sensor workflows, data stream management

1. INTRODUCTION

Different self-adaptive systems, such as smart rooms, smart offices, and smart class-rooms, have different structures and components. One common aspect of such systems, on the other hand, is that they need to process various media collected through environmental sensors and react by actuating appropriate responses. The data, which are continuously collected from the surroundings and filtered/fused by the stream processors, can be of various types, such as real-time motion, audio/video events, spatial location of objects.

In ARIA [4, 18, 19, 20], we are developing a framework to describe activities in terms of adaptive media sensing, processing, and actuating workflows. Unlike traditional workflows, a media service workflow captures

- continuous processing required at the nodes due to streaming nature of the sensory data,
- inherent redundancy and imprecision in media and alternative ways of achieving a given goal. For example, a given media object may be processed in different ways or different sensory data can be used to extract the same information. Alternatives may have different qualities (effectiveness) and costs (time and resource),
- quality of service requirements at different levels of the workflow, which is composed of individual media sensing, processing, and actuating units.

At the lowest level, ARIA media processing and integration workflows (Figure 1(a)), describe how a specific media processing task can be performed by combining various sensors, media filters and information fusion operators, and actuators. The adaptive nature of ARIA is due to the components that are programmable and adaptable: delay and quality characteristics of individual operators can be controlled via a number of parameter values.

One common characteristic of media processing applications is the *uncertainty or imprecision of data.* For example, in wireless sensor networks, the sensory data is likely to be an approximation of the physical value because of factors such as limited processing power of the data stream processor and limited battery power of sensors. In media data streams (such as video, audio and motion), raw data inputs are filtered and fused through feature extraction modules to produce a variety of feature streams. The resulting feature streams are only *statistically* accurate due to the inherent imprecision.

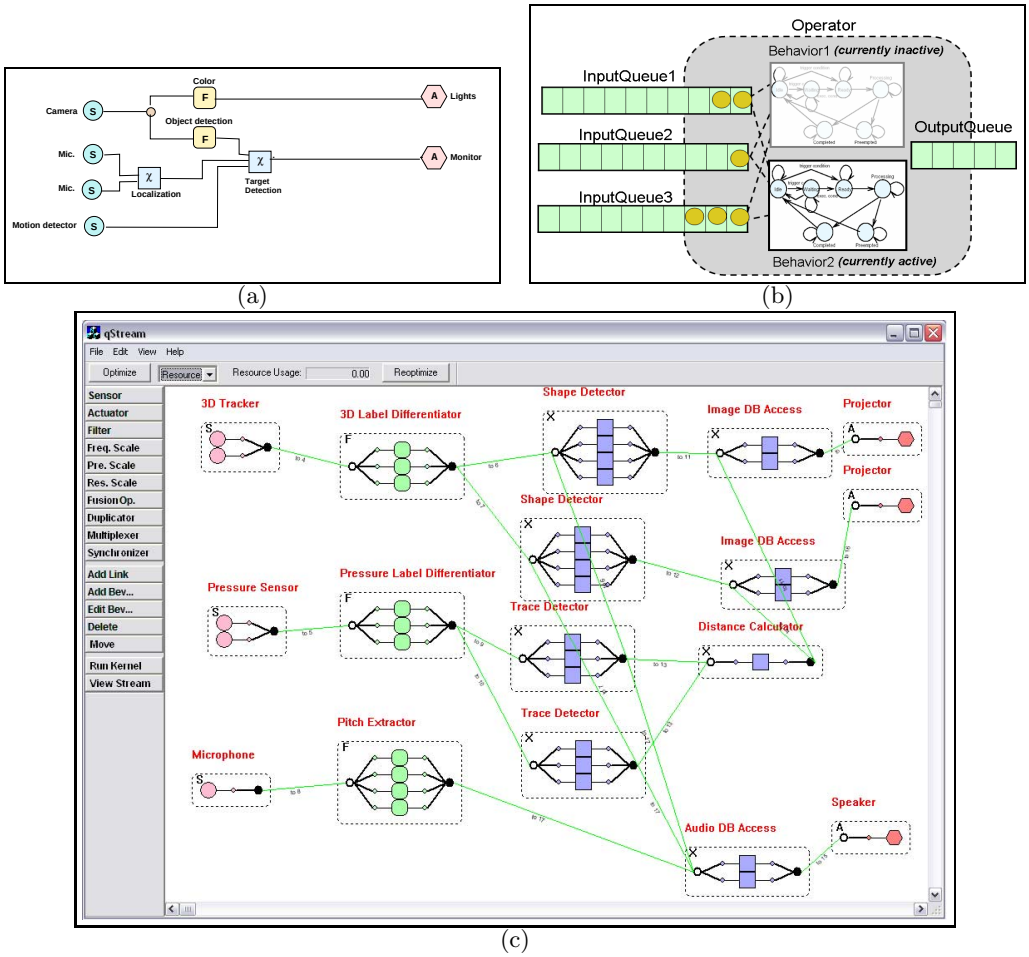

Figure 1: (a) An example media processing workflow: "S" denotes sensors, "F" denotes filters, "χ" denotes fusion operators, and "A" denotes actuators; (b) an operator with two alternative behaviors (each represented by a state machine), two input queues, and one output queue; (c) a workflow for tracking movement poses created in ARIA GUI (many processing elements in the workflow have alternative behaviors, for each such operator the most suitable behavior will need to be selected adaptively)

In [4, 20], we presented *ARIA* Media Processing Workflow Architecture, where workflow optimization was conducted to optimize the overall performance of the system in terms of output quality, resource usage, and delay, in the presence of operators that have alternative behaviors. Although the model and optimization algorithms enabled the choice of appropriate behaviors in a quality-aware manner, they did not take into account the *variabilities* in the behaviors of the operators (Figure 1(b)). In this paper, we extend the model to account for the variabilities and present a distributed optimization algorithm that takes such variability into account. We then present a quality-aware early object elimination scheme to enable informed resource savings in continuous real-time media processing workflows. In particular, stated QoS requirements, such as QoS requirements of ARIA architecture, are preregistered constraints, for instance, *"deliver data objects to actuator a, such that we can be at least κ confident that the quality of the object is above the threshold q_c"* are used for the elimination of unpromising objects from the workflows. We show that due to the adaptive nature of the ARIA workflows, choosing which objects to eliminate based on these QoS statements is not trivial. Therefore, we present an efficient and effective heuristic for real-time object selection.

2. RELATED WORK

In multimedia application management domain, Q-RAM [14, 15], provides a set of QoS optimization schemes, with discrete QoS options, in the context of video-conferencing. QoS-aware middleware, such as QOS Broker [17] and DQM [25], provides efficient QoS architectures for network transmission. ERDoS introduces the concept of dynamic application structuring to provide the best end-to-end implementation of an application in heterogeneous, distributed environments [9]. A system manager adaptively determines which resources to allocate to each application and how to schedule all applications on shared resources. Unlike ARIA, these works consider each task as an independent application. Recently, there have been a number of efforts in automatic composition of distributed multimedia services (SpiderNet [13], SAHARA [21, 22], SPY-Net [26], CANS [12], and Infopipes [3]). In most media service composition work,

the goal is to communicate a media object or a stream from a source server to a media consumer, while the overlay routing nodes provide (mostly application level) services, such as transcoding and mixing. Unlike these works, which focus on network, our focus is to address challenges associated with multimedia workflow design, adaptation, and evolution that arise in ubiquitous system design and deployment.

In data engineering domain, there has been a flurry of activities in the area of data stream and workflow management [1, 16, 11, 23, 5, 2]. Due to the continuous nature of data streams, adaptation has been a crucial aspect of these systems. On the other hand, there has been little prior work in using QoS in the context of data-flow architectures. One exception is Aurora, which focuses on QoS- and memory-aware operator scheduling and load shedding for coping with transient spikes in a data stream network [5, 24].

Previous research dealing with confidence-based queries is mainly focused on physical monitoring, feedback, and control environments, where snapshot queries are issued from the query initiator and routed over wireless sensor network to a designated node or a group of destination nodes and the destination nodes send the query result based on its reading of the physical world back to the root of the network. [10] establishes a statistical model of physical environments and augments the results of querying sensor networks with additional information on interval and the confidence. Whether a reading is acquired or not depends on whether the statistical model is sufficient to answer a query with acceptable confidence. [24] proposes a confidence model for the thermal state estimation in a wearable battlefield sensor system and the confidence level for the personal area sensor networks is determined by methods used for measuring body temperature, the latency of the data and the thermal states. The query framework in [8] is proposed for the historical data in sensor networks. [7] studied the probabilistic queries in sensor environments in which data imprecision are due to the continuous changes of physical environments and resource limitation. Imprecise data is represented by interval and a continuous probability distribution function that indicates how the real value is distributed within interval. The scope of ARIA [4, 18, 19] focused on a query architecture over the push-based media object processing workflows, where rich sensory data arrive at the processing nodes of a media processing workflow, and the results are delivered to the actuators after being processed at each operator node.

3. MEDIA PROCESSING WORKFLOWS

In ARIA framework, imperfections are due to various reasons, including the presence of media processing modules and index structures which can provide different *quality* levels depending on the available resources and processing time. The presence of a quality/resource trade-offs enable *graceful degradation* of the services provided by the system, but it also renders the *real-time* optimization and adaptation more challenging.

A particular challenge with quality-aware optimization of the media processing workflows is the *assessment* of the quality itself. If each algorithm and index-structure used in the workflow had a highly predictable *quality* (or accuracy) associated with it, the quality assessment would be straightforward. However, in general, accuracy of the algorithms themselves are unpredictable, as they depend on various factors, including the data content of the input objects being processed. Therefore, in addition to accuracy, each behavior of an operator need to be associated with a degree of confidence. Consequently, each object being streamed between the components of a media processing workflow has a *quality assessment* as well as a *degree of confidence* associated with this assessment. The basic information unit in transmission is a *data object*.

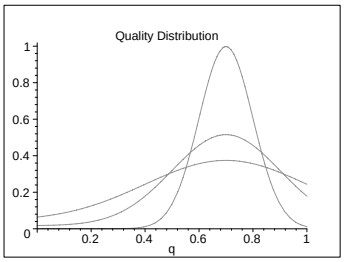

Figure 2: Example quality assessment and confidence plots with the same expected quality (0.7) but different degrees of confidence

Definition 3.1 (Object). *Each object in the system consists of:*

- *an* object payload*, such as a string, a numeric value, or an image region,*
- *an* object header*, describing the object state, including the object size, the quality assessment, degree of confidence associated with this assessment, and other meta-data.*

Figure 1(c) illustrates an example where media processing in ARIA may result in various qualities of objects depending on how the raw data is processed. In the example, two performers are scripted to draw various shapes in the air with their arms and their arm motions tracked by a 3D-motion tracking device. The output of the 3D motion tracking is filtered and clustered by a Label Differentiator. The shape detector recognizes the shapes with a degree-of-confidence and the recognized shape drives the visual content projected on the screen. Note that depending on the sampling rate, the number of 3D markers, and cameras used for identifying the exact trajectory of the performer's arm in the 3D environment, the quality of the shape labels generated can vary. Although, whichever processing alternative is chosen, the actual quality of an individual shape label depends on environmental factors (such as the performer's speed and the clarity of her movement), it is possible to assess the expected quality of a particular object as a function of the processing alternative chosen and knowledge about the environmental factors, if they are available. In this paper, we do not focus on how these assessments are computed for different media processing operations or how the operation alternatives are chosen (interested reader can refer to [6] for the former and [20] for the latter). Instead we focus on the use of these quality assessments in early pruning of unpromising objects to save resources.

Once again, note that in general it is not possible to compute the actual quality of a given object (for instance the precision of the shape label in the example given above) in real time. Instead, we use quality assessments that can

be computed based on knowledge about the media processing operators and knowledge about relevant environmental factors. Without loss of generality, we model the quality assessment of an object o and the associated confidence, using a normal distribution $N_o(q_o, \xi_o)$ (modified to fall in the range $[0, 1]$ instead of $(-\infty, \infty)$) as shown in Figure 2. The mean of the curve (0.7 in the example) denotes the quality assessment for the object, and the standard deviation denotes the *confidence* associated with this assessment: smaller standard deviation indicates higher confidence. For the validity of this model, we rely on the well-known *central limit theorem*, which states that an average tends to be normally distributed, even when the distribution from which the average is computed is not normally distributed. In fact, various statistical procedures and tools, such as *quality control charts*, rely on this to deal with process qualities.

3.1 Operator Behavior Quality Assessment

Objects are streamed between pairs of media operators in a media processing workflow.

DEFINITION 3.2 (MEDIA OPERATOR). *A media operator takes objects from its input streams and outputs a new resulting object into its output stream. Each operator is associated with a set of alternative behaviors that are* logically equivalent*, but provide different quality levels of outputs, with consuming different amount of resources (Figure 1(b)). Let v_i be an operator in the workflow,*

- *$\beta_i = \{b_{i1}, \cdots, b_{ik}\}$ is a set of behaviors associated with v_i.*
- *each behavior, $b_{i,j} : T_{i,1} \times \cdots \times T_{i,m} \rightarrow T_{i,out}$, maps m input objects to an output object. Here, $T_{i,l}$ denotes the type of the object expected at the l^{th} input queue.*
- *each behavior, $b_{i,j} \in \beta$, has an expected operation quality, also described by a normal distribution, $N_{i,j}(\varepsilon_{i,j}, \xi_{i,j})$, where $\varepsilon_{i,j}$ and $\xi_{i,j}$ denote the expected quality and the associated confidence of the behavior.*
- *each behavior, $b_{i,j} \in \beta$, has an expected delay $\Delta b_{i,j}$. In this paper, we do not focus on the variabilities in the delay of the operators.*

Media operators can be classified into two categories, based on their *operation quality* functions:

Input-independent Operators: This is the simplest case where the quality of the output object is independent of the input object or its quality assessment. In this case, the quality assessment of the output object generated by behavior $b_{i,j}$ is simply equal to $N_{out}(q_{out}, \xi_{out}) = N_{i,j}(\varepsilon_{i,j}, \xi_{i,j})$.

Input-dependent Operators. In most cases, the output quality of the operator depends on the input object qualities. In general, given a behavior, $b_{i,j}$, with only one input object, the output quality $q_{i,out}$ and confidence $\xi_{i,out}$ of the generated object can be computed as follows[1]:

[1]As we later describe in Section 3.2, in this paper we focus on so called *chain workflows*, where each operator has only one input. The problems remains hard even for this case. The computation of the output quality assessment for input-dependent operators in chain workflows is presented in the appendix. We are currently extending these results to tree as well as DAG-workflows.

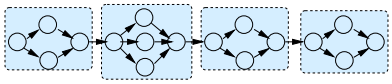

Figure 3: A chain workflow with operators with four nodes: a sensor, two intermediate filters and an actuator. Each node in the workflow has multiple behaviors

$$q_{i,out} = \varepsilon_{i,j} \times q_{in}$$
$$\xi^2_{i,out} = \xi^2_{in} \times \xi^2_{i,j} + \xi^2_{in} \times \varepsilon^2_{i,j} + q^2_{in} \times \xi^2_{i,j}$$

where q_{in} and ξ_{in} correspond to the input quality and confidence, and $\varepsilon_{i,j}$ and $\xi_{i,j}$ correspond to the operation quality and confidence of a filter. In ARIA, there are also other types of operators, where the operation quality of the operator itself is not fixed and depends on the input object or its quality assessment. For instance, the operation quality of a behavior $b_{i,j}$ may be described as $N_{i,j}(\Pi_{i,j} \times q_{in}, \xi_{i,j})$, where the expected operation quality itself depends on the input object. In general, we can denote the output object quality of such an operator as $N_{out}(f_q(q_{in}, \xi_{in}), f_\xi(q_{in}, \xi_{in}))$, for some f_q and f_ξ.

3.2 Workflows

Based on the above definitions of objects and media operators, we can now define a media processing workflow as follows:

DEFINITION 3.3 (MEDIA PROCESSING WORKFLOW). *A media processing workflow is an acyclic directed connected graph, $G(V, E)$ where*

- *V is a set of nodes that represent media operators and E is a set of edges that indicate object streams. $\forall v_i \in V$, $in(v_i)$ is the set of incoming streams and $out(v_i)$ is the set of outgoing ones.*
- *$\forall e_j \in in(v_i)$, there is a queue q_j of objects waiting for processing. If the queue length exceeds a maximum level, $size(q_i)$, then the queue manager needs to drop objects.*
- *$\forall v_i \in V$, v_i has a time window w_i such that each k-tuple of objects from different input queues of v_i received within w_i time units of each other have to be processed by a behavior, $b_{i,j}$, of node v_i, i.e. the arrival time of objects in a k-tuple, $\langle o_1, \ldots, o_k \rangle$, must be within the same time window w_i,*
- *the output frequency of the nodes are variable; in particular, at any point in time, $f_{i,out} = \frac{1}{\Delta b_{i,s}}$, where $b_{i,s}$ is the most recent selected behavior of v_i and $\Delta b_{i,s}$ is its processing delay.*

In the remainder of the paper, we focus solely on chain workflows (such as in Figure 3), where each operator has only one input. The problem of early elimination of unpromising objects remains hard even in this case, as each operator can have multiple behaviors, with different characteristics. We are currently extending these results to tree as well as DAG-workflows.

4. WORKFLOW EVALUATION WITH EARLY ELIMINATION

Various delay, resource, and quality constraints are imposed on the workflows on the *ARIA* architecture. QoS requirements of *ARIA* architecture are preregistered constraints, such as

- "deliver data objects to actuator a, such that we can be at least κ confident that the quality of the object is above the threshold q_c";

Given an object quality assessment $N_o(q_o, \xi_o)$, this constraint is equivalent to ensure that the object o satisfies the following:

$$\int_{q_c}^{1} N_o(q_o, \xi_o)\, dq \geq \kappa$$

Note that, given such a constraint, it is critical for media processing workflows to save resources by filtering out, as *early* as possible, those objects that do not satisfy this constraint. Early elimination of unpromising objects prevent promising objects from being dropped from operator queues by load shedding processes. Consequently, the early drop decisions made on the unpromising objects has a significant effect on the throughput and output quality of media processing workflows.

Consider a chain workflow (such as in Figure 3)

$$\langle op_1, op_2, \ldots, op_i, \ldots, op_n \rangle.$$

Let us also assume that an object o with quality assessment, $N_o(q_o, \xi_o)$ arrives at the input queue of operator o_i. If this object is eliminated from this queue without any processing, the total savings in the processing time (or CPU resources) this will provide is

$$save_{elim}(o, i) \leq \sum_{i \leq l \leq n} max\{\Delta b_{i,j} \mid b_{i,j} \in \beta_i\}$$

Note that an object with low current quality assessment is actually likely to consume more resources in the future to prevent further reductions in the quality. Therefore, informally, given two objects $o1$ and $o2$, such that $o1$ is less likely to satisfy the constraint than $o2$, $save_{elim}(o1, i) > save_{elim}(o2, i)$.

4.1 Early Object Elimination

Let us reconsider the object o, waiting to be processed by operator o_i. Given the remainder of the workflow $\langle op_i, \ldots, op_n \rangle$, let $p = \langle b_{i,p(i)}, \ldots, b_{n,p(n)} \rangle$ denote an execution plan (here $p(i)$, denotes the index of the chosen behavior of operator o_i). Given this plan, we can compute the combined operation quality as

$$N_{p,comb(i,\ldots,n)}(\varepsilon_{p,comb(i,\cdots,n)}, \xi_{p,comb(i,\cdots,n)}),$$

where, $\varepsilon_{p,comb(i,\cdots,n)}$ and $\xi_{p,comb(i,\cdots,n)}$ are as shown in Figure 4. Therefore, for this plan, given the object quality assessment, $N_o(q_o, \xi_o)$, we can compute an expected *final* object quality assessment, $N_{o,p,out}(q_{o,p,out}, \xi_{o,p,out})$. An execution plan, p, where

$$\int_{q_c}^{1} N_{o,p,out}(q_{o,p,out}, \xi_{o,p,out})\, dq < \kappa,$$

is not a valid plan as it does not lead into a valid final object. The set of valid plans for object o, is denoted as $P(o)$. Thus, if an object o does not have any valid plan, i.e. $P(o) = \emptyset$, the object can be trivially eliminated.

4.2 Forced Early Object Elimination

Note that, if there is at least one valid plan for o, then its elimination is not desirable. However, in some cases, an object may need to be eliminated (for example, due to the queue size constraints, even when there are valid plans for it). In these cases, we need to choose an object among all the objects in the queue such that the least possible saving from eliminating the object out of the queue is maximized. Consider an object o at the input queue of the i^{th} operator. The savings, $save_{elim}(o, i)$, from eliminating the object from the queue is lower bounded by

$$min\{\sum_{i..n} \Delta b_{i,p(i)} \mid p\ is\ a\ valid\ plan\}$$

Therefore, finding an object which has the largest minimum savings requires identifying the lowest cost valid plans for each object in the queue. However, *since there are exponentially many paths even in a chain workflow and since a shortest-path based selection of a valid path with minimum savings is not feasible*, finding such an object is not trivial. Therefore, given an object o, instead of trying to compute a valid path with the minimum savings, we use the following heuristic to find an estimate on the minimum savings (without actually identifying the corresponding path itself):

1. We first find the smallest, $save_{\perp}(o)$, and largest, $save_{\top}(o)$, possible savings. Since, we do not limit the search to valid plans, this step is trivial.
2. We compute the constraint satisfaction confidences, $conf_{\perp}(o)$ and $conf_{\top}(o)$, associated with the plans corresponding to these savings.
3. Assuming that processing delay and confidence are positively correlated, we estimate the expected valid savings, $save_{elim}(o, i)$, as

$$save_{\perp}(o) + \left(\frac{\kappa - conf_{\perp}(o)}{conf_{\top}(o) - conf_{\perp}(o)} \times (save_{\top}(o) - save_{\perp}(o)) \right)$$

Note that this is an efficient, $O(|V|klogk)$, heuristic, where $|V|$ is the number of operators on the chain and k is the maximum number of behaviors in any operator. The heuristic achieves this efficiency primarily by avoiding the computation of *valid* plans. Also, in the absence of precise knowledge about how the delay and confidence are correlated, it assumes a linear relationship between these two properties. Certainly, knowledge about this relationship would lead into better heuristics. Nevertheless, in the next section, we evaluate our confidence-based early object elimination scheme without using such extra knowledge and show that results are remarkably accurate even without relying on such information.

5. RESULTS

The goal of forced object elimination is to choose the object with the most savings in the queue for elimination. Therefore, we ran a number of experiments where we compared the *elimination ranking* obtained by a brute force search with the elimination ranking obtained by the proposed heuristic.

$$\varepsilon_{p,comb(i,\cdots,i)} = \hat{q}_{in} \times \varepsilon_{i,p(i)}$$

$$\xi^2_{p,comb(i,\cdots,i)} = \hat{\xi}^2_{in} \times (\xi^2_{i,p(i)} + \varepsilon^2_{i,p(i)}) + \hat{q}^2_{in} \times \xi^2_{i,p(i)}$$

$$\varepsilon_{p,comb(i,\cdots,n)} = \begin{cases} \varepsilon_{n,p(n)} & op_j \text{ is independent, } \forall j \in \{i,\cdots,n\}; \\ \prod_i^n \varepsilon_{j,p(j)} & op_j \text{ is dependent, } \forall j \in \{i,\cdots,n\}; \\ c_{2,n} \times \varepsilon_{p,comb(i,\cdots,n-1)} \times \varepsilon_{n,p(n)} & (c_{1,n}=1, c_{2,n}=0 \text{ if } op_n \text{ is independent,} \\ \quad +c_{1,n} \times \varepsilon_{n,p(n)} & \quad c_{1,n}=0, c_{2,n}=1 \text{ if } op_n \text{ is dependent}). \end{cases}$$

$$\xi^2_{p,comb(i,\cdots,n)} = \begin{cases} \xi^2_{n,p(n)} & op_j \text{ is independent, } \forall j \in \{i,\cdots,n\}; \\ \xi^2_{p,comb(i,\cdots,n-1)} \times (\xi^2_{n,p(n)} + \varepsilon^2_{n,p(n)}) & op_j \text{ is dependent, } \forall j \in \{i,\cdots,n\}; \\ \quad +\varepsilon^2_{p,comb(i,\cdots,n-1)} \times \xi^2_{n,p(n)} & \\ c_{2,n} \times \xi^2_{p,comb(i,\cdots,n-1)} \times (\xi^2_{n,p(n)} + \varepsilon^2_{n,p(n)}) & (c_{1,n}=1, c_{2,n}=0 \text{ if } op_n \text{ is independent,} \\ \quad +c_{2,n} \times \varepsilon^2_{p,comb(i,\cdots,n-1)} \times \xi^2_{n,p(n)} & \quad c_{1,n}=0, c_{2,n}=1 \text{ if } op_n \text{ is dependent}) \\ \quad +c_{1,n} \times \xi^2_{n,p(n)} & \end{cases}$$

Figure 4: Composite quality assessment of a chain plan p. $\hat{q}$ and $\hat{\xi}$ denote the quality assessment of the input object

We fixed the chain workflow length to 6 operators. We also varied the number of behaviors per operator from 2 to 6. The average quality, ε, of a behavior varied between 0.7 and 0.9, while the variance, ξ, was between 0.1 and 0.3. The delay of operators varied between 10 and 60 milliseconds. Note that the algorithm assumes that there is a correlation between cost and quality. In order to consider the worst case scenario for the algorithm, we did not associate any correlations between the quality- and delay-properties of the behaviors of the operators.

We considered different queue sizes (3 to 11) and used the following rank precision measure, $0 \leq rp \leq 1$, to compare brute force ranks with heuristic ranks:

$$rp = 1 - \frac{1}{k}\left(\sum_{i=1}^{k} \frac{|br_i - er_i|}{\lfloor \frac{k}{2} \rfloor + \left|\lceil \frac{k}{2} \rceil - i\right|}\right),$$

where br_i is the brute force rank of one of the k objects in the queue and er_i is its estimated rank. We ran each experiment 5 times reporting the average.

The results were very encouraging. The average precision of the algorithm for all cases was 94.1%. Even when the high precision due to cases where no-valid plan exists for any of the objects or valid plans for all objects have the same amount of savings are excluded, the rank precision of the algorithm stayed at 88.6%. As expected, for larger number of behaviors, there was a slight decrease in the ranked precision as the overall variability in the system increased in those cases, but the reduction was too slight (less than 1% in the experiments) to be of any concern.

In terms of execution time, as expected, the brute force solution proved to be unscalable in query processing time, with execution time varying from the order of milliseconds, when the number of behaviors set to 2, to the order of seconds when the number of behaviors per operator is set to 6. On the other hand, the proposed heuristic was highly scalable, with the execution time in the order of milliseconds for more complex scenarios.

6. CONCLUSION

In this paper, we first highlighted that ubiquitous and distributed media rich systems require real-time media processing workflows for filtering and fusing rich media data continuously collected from the surroundings. Since such processing is inherently resource heavy and since the data and processing both are inherently imprecise, it is important to develop workflow plans that adapt their behavior to real-time conditions. In this paper, we presented a confidence-based early object elimination scheme which ensures that resources are used only for those objects that are likely to satisfy the stated QoS constraints. We showed that due to the adaptive nature of the ARIA workflows, choosing which objects to eliminate based on these QoS statements is not trivial. Therefore, we presented an efficient and effective heuristic for object selection. We are currently extending these results to tree as well as DAG media processing workflows.

7. REFERENCES

[1] R.Avnur and J.Hellerstein. *Eddies:Continuously Adaptive Query Processing.* SIGMOD, 2000.

[2] S.Babu, *et al. Adaptive Ordering of Pipelined Stream Filters.* SIGMOD 2004.

[3] A. Black, J. Huang, R. Koster, J. Walpole and C. Pu. *Infopipes: An abstraction for multimedia streaming,* Multimedia Systems 8: 406-419, 2002.

[4] K.S. Candan, L. Peng, K.D. Ryu, K.S. Chatha and C. Mayer. *Efficient Stream Routing in Quality- and Resource-Adaptive Flow Architectures.* Multimedia Information Systems(MIS04).

[5] D.Carney*et.al. Monitoring Streams-A New Class of Data Managment Applications.*VLDB02.

[6] Yinpeng Chen and Hari Sundaram. *A Computationally Efficient 3D Shape Rejection Algorithm* Proc. ICME 2005,Amsterdam, The Netherlands, July 2005, also AME-TR-2005-05.

[7] R. Cheng, D.V. Kalashnikov and S. Prabhakar. *Evaluating Probabilistic Queries over Imprecise Data.* SIGMOD, 2003.

[8] A. Coman *et.al. A Framework for Spatio-Temporal Query Processing over Wireless Sensor Networks.* DMSN2004.

[9] Saurav Chatterjee. Dynamic Application Structuring on Heterogeneous, Distributed Systems. IPPS/SPDP Workshops 1999: 442-453

[10] A. Deshpande *et.al. Model-Driven Data Acquisition in Sensor Networks.* VLDB, 2004.

[11] D.J.DeWitt *et al. Gamma - A High Performance Dataflow Database Machine.* VLDB 1986.

[12] X. Fu, W. Shi, A. Akkerman and V. Karamcheti. *CANS: Composable, Adaptive Network Services Infrastructure,* Proc. of USITS, March, 2001.

[13] X. Gu and K. Nahrstedt. *Distributed Multimedia Service Composition with Statistical QoS Assurances,* IEEE Transactions on Multimedia, 2005.

[14] C. Lee, J. Lehoczky, R. Rajkumar and D. Siewiorek. *On Quality of Service Optimization with Discrete QoS Options,* IEEE RTSS, 1999.

[15] C. Lee, J. Lehoczky, D. Siewiorek, R. Rajkumar and J. Hansen. *A Scalable Solution to the Multi-Resource QoS Problem,* IEEE RTSS, 1999.

[16] S.Madden, M.Shah, J.Hellerstein and V.Raman. *Continuously Adaptive Continuous Queries over Streams.* SIGMOD, 2002.

[17] K. Nahrstedt and J. M. Smith. *The QOS Broker,* IEEE MultiMedia, Vol. 2, Issue 1: 53 - 67, 1995.

[18] L. Peng, K.S. Candan, K.D. Ryu, K.S. Chatha and H. Sundaram. *ARIA: An Adaptive and Programmable Media-flow Architecture for Interactive Arts.* ACM MM Arts Program, 2004.

[19] L. Peng, K. S. Candan, *Adaptive Multi-Sensor, Multi-Actuator Media Workflow System for Interactive Arts.* Workshop on Managing Data for Emerging Multimedia Applications(EMMA05).

[20] L. Peng, K.S. Candan, C. Mayer, K.D. Ryu and K.S. Chatha. *Optimization of Media Processing Workflows with Adaptive Operator Behaviors.* to appear in Multimedia Tools and Applications, Kluwer.

[21] B. Raman and R. H. Katz. *An architecture for highly available wide-area service composition,* Computer Communication, 26(15):1727–1740, September 2003.

[22] B. Raman and R. H. Katz. *Load Balancing and Stability Issues in Algorithms for Service Composition,* IEEE INFOCOM, 2003.

[23] M.A.Shah and S.Chandrasekaran, *Fault-Tolerant, Load-Balancing Queries in Telegraph,* SIGMOD Record, v.30 n.2, p.611, June 2001.

[24] N. Tatbul *et.al. Confidence-based Data Management for Personal Area Sensor Networks.* DMSN2004.

[25] N. Wang and G. Pavlou. *DQM : An Overlay Scheme for Quality of Service Differentiation in Source Specific Multicast,* workshop on QoS-IP, 2003.

[26] D. Xu and K. Nahrstedt. *Finding Services Paths in a Media Service Proxy Network,* MMCN, 2002.

Appendix: Output Quality Assessment for Input-Dependent Operators

As discussed in Section 3.1, in many cases, the output qualities of operators depend on the input object qualities. Since in this paper, we focus on chain workflows, in this appendix, we describe how to compute the output quality $q_{i,out}$ and confidence $\xi_{i,out}$ of the generated object, given a behavior, $b_{i,j}$, with only one input object. In this appendix, we use the following notation:

- q is a variable associated with the object qualities.
- t is a variable associated with the operator qualities.
- q_{in} is the mean of the quality assessment of the input.
- ξ_{in} is the confidence level for estimating the quality q_{in} of the input object (i.e., the standard deviation with respect to the mean; this measures the spread in the quality of the input object).
- $\varepsilon_{i,j}$ is the mean operation quality of the selected operator.
- $\xi_{i,j}$ is the confidence level for estimating the operation quality $\varepsilon_{i,j}$ (once again, this is the standard deviation with respect to the mean and measures spread in the operation quality of the behavior).
- q_1 is the first moment about the origin of the object quality variable q (is equal to q_{in}).
- t_1 is the first moment about the origin of the operator quality variable t (is equal to $\varepsilon_{i,j}$).
- q_2 is the second moment about the origin of the object quality variable q.
- t_2 is the second moment about the origin of the operator quality variable t.
- $q_{i,out}$ is the mean quality of the output objects that are generated by node i given q_{in}.
- $\xi_{i,out}$ is the confidence for estimating the quality $q_{i,out}$ of the output objects (i.e., the standard deviation).

Consider an input-dependent operator with the window size of one object that takes an input object from the input queue and outputs a new object. Let the random variables q and t respectively, denote the input quality and operation quality. Assume that q and t are independent and normally distributed in the range $[0, 1]$. The quality of the new object is a random variable given by $q \times t$. The expected quality of the new object is then given by:

$$\begin{aligned} q_{i,out} &= \int_0^1 \left(\int_0^1 t \times N_{i,j}(\varepsilon_{i,j}, \xi_{i,j})\, dt \right) \\ &\qquad \times q \times N_{in}(q_{in}, \xi_{in})\, dq \\ &= \int_0^1 \varepsilon_{i,j} \times q \times N_{in}(q_{in}, \xi_{in})\, dq \\ &= \varepsilon_{i,j} \times q_{in} \end{aligned}$$

Thus, the expected quality of the new object is the product of the expected quality of input objects and the expected operation quality of operators which conforms to intuition.

The variance of the product then can be computed as follows:

$$\begin{aligned} \xi^2_{i,out} &= \int_0^1 \left(\int_0^1 (q \times t - q_{i,out})^2 \times N_{i,j}(\varepsilon_{i,j}, \xi_{i,j})\, dt \right) \\ &\qquad \times N_{in}(q_{in}, \xi_{in})\, dq \\ &= \int_0^1 \left(\int_0^1 (q^2 t^2 - 2qtq_{i,out} + q^2_{i,out}) \times N_{i,j}(\varepsilon_{i,j}, \xi_{i,j})\, dt \right) \\ &\qquad \times N_{in}(q_{in}, \xi_{in})\, dq \\ &= q_2 t_2 - 2q_1 t_1 q_{i,out} + q^2_{i,out} \\ &= q_2 t_2 - 2q^2_{i,out} + q^2_{i,out} \\ &= q_2 t_2 - q^2_{i,out} \\ &= (\xi^2_{in} + q^2_{in})(\xi^2_{i,j} + \varepsilon^2_{i,j}) - (\varepsilon_{i,j} \times q_{in})^2 \\ &= \xi^2_{in} \times \xi^2_{i,j} + \xi^2_{in} \times \varepsilon^2_{i,j} + q^2_{in} \times \xi^2_{i,j} \end{aligned}$$