Efficient Dependency Tracking in Packetised Media Streams

Alexander Eichhorn
Distributed Systems and Operating Systems Group
Ilmenau Technical University, P.O. Box 100 565, 98684 Ilmenau, Germany
Email: alexander.eichhorn@tu-ilmenau.de

***Abstract*— Scheduling and error control mechanisms for robust delivery of media streams over packet networks rely on distortion metrics to optimally allocate resources and protect streams from uncontrolled quality degradation. Current distortion metrics are accurate, but the actual distortion values are expensive to obtain. Therefore, distortion models often assume fixed dependency patterns and neglect fragmentation issues. While this decreases runtime complexity, it also limits the application of such models to special stream classes and network environments.**

In response, we present a practical, efficient and format-independent framework to reason about dependencies in media streams. Based on correlation analysis we show that the estimations made by our framework match traditional distortion metrics for a number of H.264 encoded streams. Performance benchmarks indicate, that our framework is applicable at very low computational overheads.

I. Introduction

Protocols for the robust delivery of media streams over best-effort packet networks require a notion of data importance to efficiently protect data units from transmission errors. Previous work on content-aware protocols has shown that the quality of reconstructed signals at the receiver can be significantly improved when information about expected channel errors and the expected distortion with respect to the loss of data units is available at encoding or transmission time.

Based upon this information, protocols may perform selective drop [1], [2], packet interleaving [3], unequal error protection and forward error correction [4]–[6], packet scheduling and selective retransmissions [7]–[9] as well as priority mappings to Quality-of-Service classes supported in certain network environments [10], [11].

Common to these approaches is that they express distortion in metrics defined in the signal domain, either directly measured as the mean squared error (MSE) between a reconstructed signal and a reference signal, or by approximating the expected distortion based on models [9], [12]–[14].

Although distortion metrics are known to yield accurate importance estimations, obtaining distortion values is expensive. It requires multiple channel simulations and is limited to pre-encoded content. Distortion models on the other hand often assume special encoding and concealment schemes, fixed dependency structures, and a one-to-one mapping between encoded data units and transport units. This limits their utility in multicasting and broadcasting scenarios, conflicts with advanced video coding features as well as packet fragmentation in real systems.

In this paper we present an efficient and universal dependency tracking framework for packetised media streams that is independent of encoding schemes and stream structures. Our framework is based on static type descriptions and dynamic dependency graphs that can be efficiently traversed at run-time.

Our framework can be used (1) in rate-distortion models and transport mechanisms to reason about actual dependencies and (2) in streaming protocols as a low-complexity alternative to traditional distortion metrics for estimating the importance of data units. This framework extends our previous work on dependency modelling [15] by the support for fragmented streams and different fragmentation semantics, such as data-partitioning and multiple description coding.

Although dependency-based importance is less accurate than distortion metrics, we show by correlation analysis that the importance estimations made by our framework match to MSE distortion and SSIM distortion [16] for a number of H.264 encoded streams. We also provide performance benchmarks to demonstrate that the reduced complexity of a graph-based dependency model allows us to compute the importance of 1800 parallel H.264 streams in realtime at one percent load on recent processors.

Using graphs to represent dependency relations is common practice. As applied in our framework, inter-frame dependencies can be modelled as a directed acyclic graph (DAG) [17], [18] or as partially ordered set [1], [19], whereas both notations are convertible. Röder et. al. [18] uses graphs to extend the work of [9] by efficient algorithms that generate more precise policy vectors for the RaDiO framework, but because of their close integration, the graphs are not universally applicable.

While most protocols implicitly use static IPB-dependency schemes, few authors consider more complex and variable dependency structures, such as [19] for dependency-aware transmission-order scheduling. Some approaches deduce the importance of data units directly from their types [11], [20]. While such static classification schemes are state-less and easy to implement, they lack history features and cannot predict the importance of lost data units. In contrast, our framework is specifically designed to reason about these dynamic properties. To the best of our knowledge it is the first that solely considers types and dependency relations between data units for importance estimation and structural analysis.

In the remainder of this paper we first introduce the main concepts of dependency modelling. Next, we present

a comparative analysis between traditional distortion metrics and the importance estimations obtained from our model. Finally, we present performance benchmarks of a prototype implementation.

II. Dependency Modelling

The intention of our dependency model is to provide a universal and practical tool to infer the dependency-based importance of data units in media streams (even that of lost units) and to analyse streams for broken dependency that occurs when referenced data units are lost or dropped. Broken dependency directly translates into error propagation and poor quality during signal reconstruction and thus resilient media streaming protocols aim to constrain these effects.

We assume that a static type description is reliably exchanged prior to the stream. This description contains static dependency information about all data unit types. It must be specified once per encoding format using a dependency description language, which is part of our framework (see [15] for details). The description is sufficiently small (a few bytes) to be included in media signalling protocols.

We also assume that each data unit is properly fragmented by the application or transport protocol, but we do not restrict the model to a particular packetisation scheme, such as frames, slices, NAL units or RTP packets. We further assume that either type information or dependency hints are attached to data units using format-independent labels (see table I).

Purpose	Attribute	Description
Strictly pred. streams	`seq`	unique sequence number
Limitedly predictable streams	`type`	data unit type
	`epoch`	dependency epoch
	`enclayer`	encoding layer (optional)
	`reflayer`	referenced layer (optional)
Unpredictable streams	`short_term_reflist`	seq. of short-term references
	`long_term_reflist`	seq. of long-term references
	`is_long_term_ref`	marks long-term references
Group-based streams	`group_seq`	in-group position
	`group_size`	data units in this group
	`imp_boost`	additional importance boost

TABLE I
Attributes in data unit labels.

Because our model only considers dependencies it is unaware of any signal processing (error concealment) at the receiver. We do not assume that senders and receivers have perfect knowledge about the future of a stream. However, the model may be able to predict future structure and dependencies when the encoding format is properly defined.

Our dependency model does only require limited storage to cache information about data units within the window of interest for a particular protocol and it introduces no conceptional delay to stream processing. Hence, it is applicable to streaming of pre-encoded content as well as live streaming scenarios over typical packet networks where data units are reordered and lost.

A. Data Model and Dependency Representation

For representing actual dependencies between data units we use a directed acyclic graph, the object graph G_O, while the static dependency structure of a particular stream format is expressed in the type graph G_T, that may contain loops or cycles. The object graph is generated dynamically when data units become visible, while the type graph is constructed from the static type description.

The object graph keeps record of the set of visible data units and the short-term history of the stream. This is sufficient, because most of the dependency relations are short-term, that is, the distance between directly and transitively depending data units is limited by the encoding format. We use the concept of *epochs* to express boundaries that limit transitive dependency relations to continuous stream sections. A prominent example for epochs is the group-of-pictures (GOP) concept. Epochs are not required to have a fixed size or a pre-defined dependency pattern throughout the stream. Instead we allow flexible epochs to express the dynamic dependency modes of advanced video coding schemes.

B. Visibility and Predictability

In real systems the visibility of data units is limited, either because a stream is generated live or data units are intentionally dropped, reordered or lost in transit. A sender is only aware of data units that have been already generated while a receiver can only have information about properly received units. Type and dependencies of lost and future units is invisible. Invisibility directly influences estimation accuracy because dependency relations may be concealed. Hence, it is desirable to predict importance or dependencies.

While generally impossible, prediction is feasible when streams possess fixed-size epochs and regular dependency patterns. We call such streams *strictly predictable*. Other streams may have *limited predictability*, because their epoch size varies, but their dependency is reconstructable from types. However, most streams are regarded *unpredictable*, because they contain variable references, variable numbers of fragments per data unit, or dependency-independent type systems.

C. Object Graph Decoration

Decoration is the process of inserting a new, lost, reordered or repaired data unit into the object graph. The kind of decoration depends on the available or recoverable information: For strictly and limitedly predictable streams sequence number, type and epoch are sufficient. In combination with static type information *implicit decoration* can insert all relations. This works even for lost units when their type is predictable. For unpredictable streams, however, more expensive dependency lists and *explicit decoration* are required. With explicit dependencies, lost data units can also accumulate references even if their type is unpredictable. Decoration can immediately detect a *broken dependency* due to type mismatch, the selection of a lost data unit or any of its transitive successors as target. Broken dependency is only resolved when any lost predecessor is successfully updated.

D. Dependency Analysis and Importance Estimation

Based on the object graph structure, our model can perform queries, structure validation and importance estimation. Queries are useful in protocols to analyse if data units are transitively reachable from other units, or which units are affected by the loss of one or more units. Validation checks if all required dependency relations for a given data unit u are satisfied and if no unit in the backward transitive closure of u in G_O exhibits a broken dependency.

Importance estimation recursively determines the number and importance values of all depending units. In contrast to validation, estimation visits the forward transitive closure of a data unit. The recursion ensures that importance values increase monotonically over any visible dependency chain.

E. Groups of Data Units

So far, we intuitively used dependencies to describe that the existence of one data unit is *essential* for processing a related unit. This is sufficient when data units contain complete frames, but fragmentation, data partitioning and multiple description coding require different semantics. Therefore we introduce the concept of groups and define group semantics to specify a particular relationship inside a group.

a) Equal containment groups: model data units that are fragments of a larger application-level unit. Transport protocols may use them to express dependencies after fragmentation. Our model considers a group member valid if the group is complete and assigns equal importance values to all members.

b) Unequal containment groups: model data units in layered, data partitioned, FEC-encoded or otherwise unequally important streams. Our model assigns each group member a different importance value, relative to the importance of the group. A data unit is valid if all group members with a relative importance larger than a type-specific threshold are available.

c) Refinement groups: express relations in multiple description coded streams, where at least a single (but arbitrary) member is required for decoding of a basic quality signal. Our model raises importance values above the group importance when only a few group members are available. Refinement group members are instantly valid if they are not lost.

Group semantics are a property of the data unit type, but groups may span data units of multiple types. As a practical example, consider data partitioned H.264 streams, where each slice is segmented into NAL units of type 2, 3, and 4, and all NAL units containing data from a single frame become members of a single unequal containment group.

III. Evaluation

In order to evaluate the accuracy and performance of our dependency model we implemented a prototype system in C++ and performed statistical analysis and benchmarks.

For a maximum variability in stream features we selected standard sequences with different frame sizes and different motion characteristics. We also used a longer non-standard sequence, obtained from the BBC Motion Gallery[1]. All sequences were encoded at different settings using the H.264 reference encoder (JM12)[2] and the open-source x264 encoder[3] (see table II for properties of the encoded streams). We selected four dependency patterns: (1) a fixed I-P-B structure (IPB), (2) a fixed 3-layer pyramid structure (Pyramid), (3) x264 adaptive reference picture selection with B-Frames enabled (Adaptive), and (4) JM12 error-resilient slice packtisation mode with intra-refresh and adaptive reference picture selection (Resilient). While the x264 encoder can only generate one slice per frame, we configured the JM12 encoder for pattern 4 to create NAL units with maximal size of 1450 bytes. All sequences have a GOP size of 24, contain one IDR-frame per GOP to hard-limit error propagation. Pattern 4 is encoded with forced intra-refresh of one line of macroblocks per frame.

Name	Frame-size	Bitrate (kbit/s)	x264 units	JM12 NAL units (per frame) total	mean	std	max
Akiyo	176x144	300	327	632	2.092	1.486	9
Coastguard	352x288	300	327	595	1.970	1.634	8
Foreman	352x288	300	327	588	1.947	1.765	11
CBSNews	720x576	600	5651	19301	3.722	6.279	129

TABLE II
PROPERTIES OF VIDEO STREAMS: X264 USES ONE NALU PER FRAME, AND JM12 MATCHES NALU SIZE TO NETWORK PACKETS. TOTAL: OVERALL NUMBER OF NALUS; MEAN/STD/MAX: PER-FRAME STATISTICS

A. Correlation Analysis

To figure out how close our estimated dependency-based importance matches traditional distortion metrics we performed a statistical analysis over different streams and dependency structures. To obtain the expected distortion values with respect to the loss of a NALU we systematically removed each NALU from a stream and calculated MSE and SSIM metrics [16] between the decoded version of the resulting sequences and their original, error-free versions.

For sound statistical analysis we removed extreme outliers that resulted from decoder crashes and applied the Box-Cox transformation [21] to normalise the data. We then calculated the Spearman rank order correlation coefficient and the Pearson correlation coefficient. The results in table III show good correlation in both distortion metrics when the distortion is generated by a lost frame (all x264 streams). This indicates that dependencies have a major impact on error propagation, even for adaptive and weighted reference frame selection.

If, however, distortion is isolated to smaller sections of a frame and error propagation is stopped faster due to intra updates (as for JM12 streams), the dependency-based importance matches poorly with real distortion. We suspect our model to over-estimate importance in such cases because it only considers dependencies between groups of data units. All data units in a group are treated equal although although a unit's

[1] http://www.bbcmotiongallery.com/Customer/Showreels.aspx
[2] http://iphome.hhi.de/suehring/tml/
[3] http://www.videolan.org/developers/x264.html

contribution to signal quality decreases with increased frame and group size. A useful extension to improve its performance would be to attach weights to vertices in the dependency graph that represent the amount of error propagation and error correction caused by skipped/predicted/intra macroblocks contained in a data unit.

Pattern	Stream	MSE SROCC	MSE CC	MSE OR	SSIM SROCC	SSIM CC	SSIM OR
x264 IPB	Akiyo	0.821	0.735	0.092	-0.570	-0.566	0.092
	Coastguard	0.891	0.846	0.092	-0.719	-0.650	0.092
	Foreman	0.844	0.783	0.092	-0.422	-0.400	0.092
	CBS News	0.651	0.603	0.083	-0.364	-0.390	0.083
x264 Pyramid	Akiyo	0.741	0.726	0.092	-0.728	-0.834	0.092
	Coastguard	0.750	0.735	0.092	-0.706	-0.734	0.092
	Foreman	0.730	0.722	0.092	-0.584	-0.569	0.092
	CBS News	0.575	0.581	0.108	-0.434	-0.463	0.108
x264 Adaptive	Akiyo	0.540	0.572	0.089	-0.049	-0.113	0.089
	Coastguard	0.891	0.848	0.089	-0.056	-0.056	0.089
	Foreman	0.718	0.705	0.089	-0.143	-0.152	0.089
	CBS News	0.668	0.606	0.083	-0.390	-0.401	0.083
JM12 Resilient	Akiyo	0.305	0.345	0.004	-0.225	-0.356	0.004
	Coastguard	0.335	0.337	0.001	0.005	0.027	0.001
	Foreman	0.296	0.282	0.001	0.096	0.103	0.001
	CBS News	0.300	0.308	0.001	-0.091	-0.028	0.001

TABLE III
PERFORMANCE COMPARISON OF IMPORTANCE ESTIMATION MODELS. CC: CORRELATION COEFFICIENT; SROCC: SPEARMAN RANK ORDER CORRELATION COEFFICIENT; OR: OUTLIER RATIO

B. Performance Benchmarks

Figure IV shows the runtime costs of graph decoration and importance estimation algorithms. The measurements were performed on a 64bit Intel Core2Duo processor with 2.16 GHz and 4MB 2^{nd}-level cache. Each sample represents the statistical mean of a single call measured over all data units in the particular streams.

As expected, the runtime costs for both functions are low. They slighty increase when group sizes grow larger. The figures indicate that our model scales well with group sizes and that it is able to track and analyse up to 1800 'CBS News' streams at one percent CPU usage. Due to structure validation checks, performed when inserting new data units, the decoration function is more expensive than importance estimation. Because estimations may be required multiple times in real protocols, we optimised for this case.

IV. CONCLUSION

We proposed a universal framework for dependency tracking and importance estimation in multimedia streams that utilises low-complexity dependency graphs as an alternative to complex distortion metrics. Our framework can serve as enhancement to existing distortion models or as a tool to directly obtain sufficiently accurate importance values. We expect this framework to be valuable in situations where distortion is unavailable or too expensive such as in realtime scenarios, for resource-constrained devices, and in transport protocol mechanisms that operate at the packet-level.

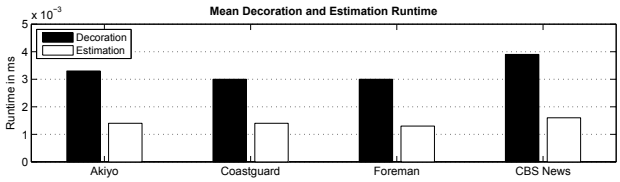

Fig. 1. Performance Benchmarks for Decoration and Importance Estimation.

REFERENCES

[1] C. Krasic and J. Walpole, "Quality-adaptive Media Streaming by Priority Drop," in *Proc. of NOSSDAV '03*, 2003.

[2] W. Tu and J. C. E.Steinbach, "Rate-distortion optimized frame dropping and scheduling for multi-user conversational and streaming video," *J. of Zhejiang University*, vol. 7, no. 5, 2006.

[3] J.-Y. Choi and J. Shin, "A novel content-aware interleaving for wireless video transmission," *Computer Communications*, vol. 29, no. 13-14, pp. 2634–2645, 2006.

[4] J. Kim, R. Mersereau, and Y. Altunbasak, "Distributed video streaming using multiple description coding and unequal error protection," *IEEE Trans. on Image Processing*, vol. 14, no. 7, 2005.

[5] M. van der Schaar and H. Radha, "Unequal Packet Loss Resilience for Fine-Granular-Scalability Video," *IEEE Trans. on Multimedia*, vol. 3, no. 4, pp. 381–393, 2001.

[6] H. Cai, B. Zeng, G. Shen, , and S. Li, "Error-Resilient Unequal Protection of Fine Granularity Scalable Video Bitstreams," in *Proc. of ICC 2004*, June 20-24 2004.

[7] M. G. Podolsky, S. McCanne, and M. Vetterli, "Soft ARQ for Layered Streaming Media," *Journal on VLSI Signal Processing Systems*, vol. 27, no. 1-2, pp. 81–97, 2001.

[8] C.-M. Chen, C.-W. Lin, and Y.-C. Chen, "Packet scheduling for video streaming over wireless with content-aware packet retry limit," in *Workshop on Multimedia Signal Processing*, 2006.

[9] P. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," *IEEE Trans. on Multimedia*, vol. 8, no. 2, 2006.

[10] J. Shin, J. Kim, and C.-C. J. Kuo, "Quality of service mapping mechanism for packet video in differentiated services network," *IEEE Trans. on Multimedia*, vol. 3, no. 2, pp. 219–231, June 2001.

[11] V. Lecuire, F. Lepage, and K. Kammoun, "Enhancing quality of mpeg video through partially reliable transport service in interactive application," in *Proc. of MMNS '01*, 2001.

[12] Y. Eisenberg, F. Zhai, T. P. Berry, and A. Katsaggelos, "VAPOR: Variance-Aware per-pixel Optimal Resource Allocation," *IEEE Transactions on Image Processing*, vol. 15, no. 2, pp. 289–299, 2006.

[13] Y. Liang, J. Apostolopoulos, and B. Girod, "Analysis of packet loss for compressed video: Does burst-length matter," in *Proc. of IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 2003.

[14] J. Chakareski, J. Apostolopoulos, S. Wee, W. Tan, and B. Girod, "Rate-distortion hint tracks for adaptive video streaming," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 15, no. 10, 2005.

[15] A. Eichhorn, "Modelling Dependency in Media Streams," in *Proc. of ACM Intl. Conf. on Multimedia*, 2006, pp. 941 – 950.

[16] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[17] G. Cheung and W.-T. Tan, "Directed Acyclic Graph based Source Modeling for Data Unit Selection of Streaming Media over QoS Networks," in *Int. Conf. Multimedia & Expo*, 2002.

[18] M. Röder, J. Cardinal, and R. Hamzaoui, "Branch and bound algorithms for rate-distortion optimized media streaming." *IEEE Trans. on Multimedia*, vol. 8, no. 1, pp. 170–178, 2006.

[19] S. Varadarajan, H. Q. Ngo, and J. Srivastava, "Error Spreading: A Perception-Driven Approach to Handling Error in Continuous Media Streaming," *IEEE/ACM Trans. on Networking*, vol. 10, no. 1, 2002.

[20] U. Choudhry and J. Kim, "Performance evaluation of h.264 mapping strategies over ieee 802.11e wlan for robust video streaming," in *Pacific-Rim Conf. on Multimedia*, 2005.

[21] G. E. P. Box and D. R. Cox, "An analysis of transformations," *Journal of the Royal Statistical Society*, pp. 211–243, discussion 244–252, May 1964.