Load Balance Properties of Distributed Data Layouts for Clustered MOD Servers *

Milind M. Buddhikot, Guru Parulkar

WUCS-95-32

May 95

Department of Computer Science
Campus Box 1045
Washington University
One Brookings Drive
St. Louis, MO 63130-4899

*This work was supported in part by the ARPA, National Science Foundation, and an industrial consortium of Ascom Timeplex, Bellcore, BNR, Goldstar, NEC, NTT, SynOptics, and Tektronix.

Load Balance Properties of Distributed Data Layouts for Clustered MOD Servers

Milind M. Buddhikot
milind@dworkin.wustl.edu
small +1 314 935 4203

Gurudatta M. Parulkar
guru@flora.wustl.edu
+1 314 935 4621

Abstract

Large scale storage servers that provide location transparent, interactive access to hundreds or thousands of concurrent, independent clients will be important components of the future information super-highway infrastructure. Two key requirements of such servers are as follows: support high parallelism and concurrency in data access to allow large number of access to the same or different data. Second, support independent interactive playout control operations such as *fast-forward, rewind, slow-play, pause, resume, random access etc.* with minimal latency. This paper assumes a distributed storage server architecture consisting of several high performance storage nodes interconnected by a high speed desk area network into a cluster as a candidate architecture that can meet these two requirements. For such an architecture, we explore generalized distributed data layouts to satisfy the requirement of large number of scalable concurrent data accesses. We also quantify certain interesting properties of these data layouts that guarantee efficient implementation of interactive operations.

1. Introduction

Large scale storage servers that provide location transparent, interactive access to hundreds or thousands of concurrent, independent clients will be important components of the future information super-highway infrastructure. MIMD and SIMD Multiprocessor supercomputer based machines have been considered as candidate architectures for such high performance servers [2, 3, 1]. However, these server architectures tend to be specialized, expensive and may prove to be overkill. An alternate architecture shown in Figure 1, commonly called as Clustered Multimedia Storage (CMS), has been proposed by several research groups including ours [8, 4, 5, 14]. It essentially consists of a set of independent storage nodes that are connected together by a fast packet based interconnect such as a packet switched bus, a ring or even a general purpose multicast switch. The interconnect directly interfaces to an external high speed network such as an ATM architecture.

Figure 2 illustrates an example CMS architecture that uses a novel ATM Port Interconnect Chip (APIC) based 2.4 Gbps cell swicthed interconnect and high performance storage nodes constructed

This work was supported in part by the ARPA, National Science Foundation, and an industrial consortium of ascom Timeplex, Bellcore, BNR, Goldstar, NEC, NTT, Southwestern Bell, SynOptics, and Tektronix.

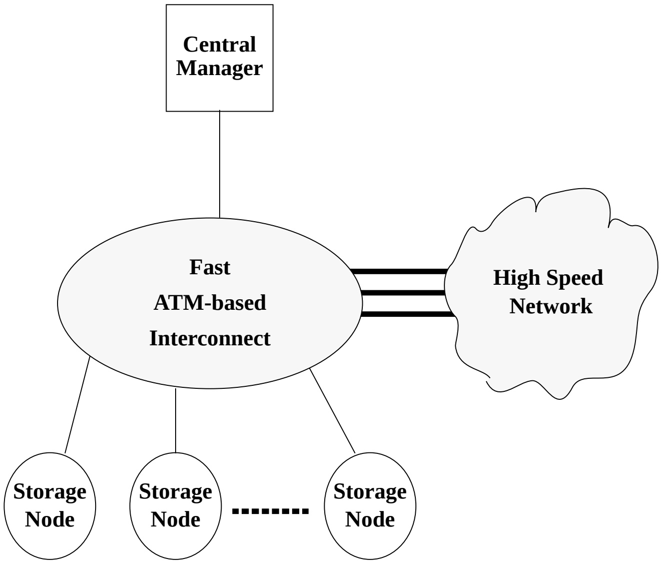

Figure 1: Clustered Multimedia Storage Server

out of Pentium PCs. This architecture is currently being prototyped at the Washington University in St. Louis under the NSF's National Challenge Award (NCA) grant aimed at deploying a scalable Multimedia-On-Demand (MOD) server. Details of this architecture can be found in [8].

Two important performance metrics of such any large scale server are **parallelism** and **concurrency** [9]. The parallelism (P_f) metric is defined as the number of storage devices simultaneously participating in supplying data for a document being accessed by a connection. Large parallelism increases network and storage throughput and thus, increases scalability. On the other hand, the concurrency C_f metric is defined as the number of clients that can simultaneously access the same document f. Ideally, maximum number of clients should be served from a single copy. In other words, higher concurrency minimizes need for copy replication. Clearly, P_f and C_f are related; increasing P_f increases C_f. Thus, key to achieving scalability is to increase parallelism and concurrency. The parallelism can be increased by physically distributing data over multiple storage nodes.

We use these key observations to devise data layouts called *Generalized Staggered Distributed Cyclic* (G-SDCL) data layouts. In these layouts, the data for bandwidth internsive streams such as video, graphics, animation documents are physically striped over multiple storage nodes, whereas the data for streams such as audio, text, data that are less bandwidth intensive are however, confined to a single storage node.

The remainder of this paper concerns with various issues in the design of such data layouts and is organized as follws. Section 2 describes the basic Generalized Staggered Distributed Cyclic layout and motivates the need for certain load-balance properties such layouts must posses to allow efficient implementation of interactive operations such as *ff* and *rw* and defines the "safe skipping distance" (d_f, d_r) for implementation of such operations. Section 3 develops the theorems that characterize the safe values for d_f, d_r for G-SDCL$_0$, G-SDCL$_1$, G-SDCL$_{k_s}$ layouts. Section 4 provides a brief discussion of implication of non-unit chunk size on load-balance properties and implementation of *ff* and *rw* operation. In Section 5 we present related work. Finally, Section 6 presents the conclusions and on-going work

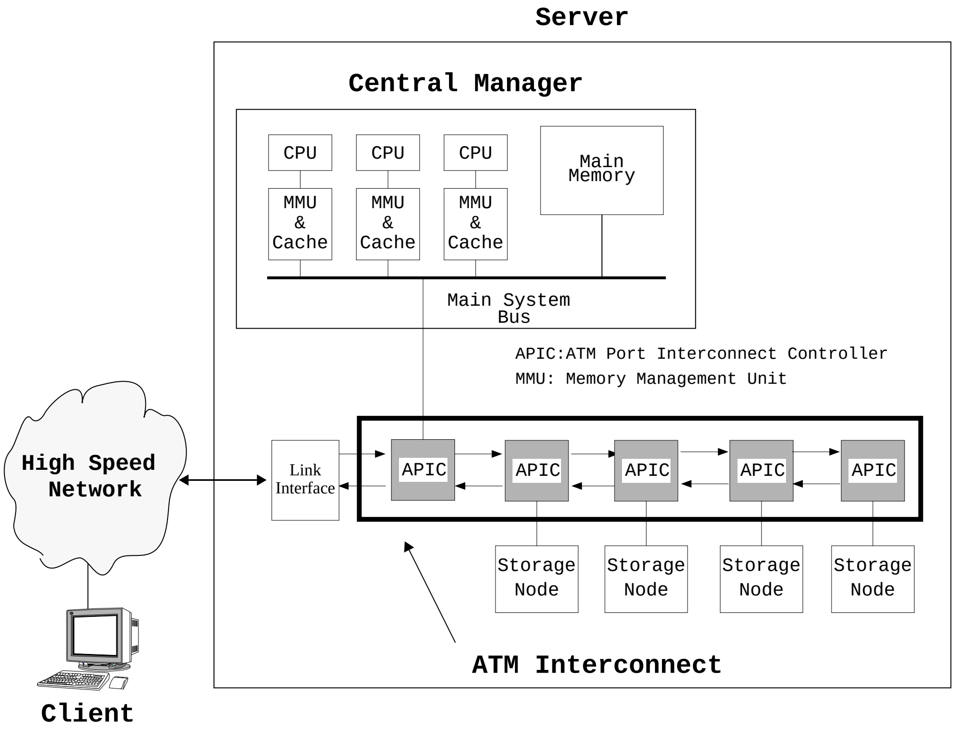

Figure 2: A prototype implementation of a CMS server

2. Distributed Data Layouts

This section first discusses the basic Generalized Staggered Distributed Cyclic Layout (GSDCL) and then, motivates the need for investigating of *"load-balance properties"* of such layouts that are crucial to efficient implementation of interactive operations such as *fast-forward* and *rewind*. We assume that these operations are implemented by keeping the display rate constant and skipping frames, where the number of frames to skip is determined by the *ff* rate and the data layout. Thus, *ff* may be realized by displaying every alternate frame, every 5^{th} frame, or every d^{th} frame in general.

2.1. Generalized Staggered Distributed Cyclic Layout (GSDCL)

We use the fact that the multimedia data is amenable to spatial striping to distribute it hierarchically over several autonomous storage nodes within the server. Figure 3 illustrates the a data layout called *Generalized Staggered Distributed Cyclic Layout* (G-SDCL). It uses The layout uses a basic unit called *"chunk"* consisting of k consecutive frames. All the chunks in a document are of the same size and thus, have a constant time length in terms of playout duration. In case of a Variable Bit Rate (VBR) video such as MPEG video, a chunk therefore represents a Constant Time Length (CTL) but a variable data length unit. In case of a Constant Bit Rate (CBR) source, it also has constant size [10, 13]. Different documents may have different chunk sizes, ranging from $k = 1$ to $k = F_{max}$, where F_{max} is the maximum number of frames in a multimedia document. In case of MPEG compressed streams, the group-of-pictures (GOP) is one possible choice of chunk size. A chunk is always confined to one storage node. The successive chunks are distributed over storage nodes using a logical layout topology. For example, in Figure 3 the chunks have been laid out using a ring topology. Each such ring is called a *distribution cycle*. The layout can thus be thought of as a succession of such cycles

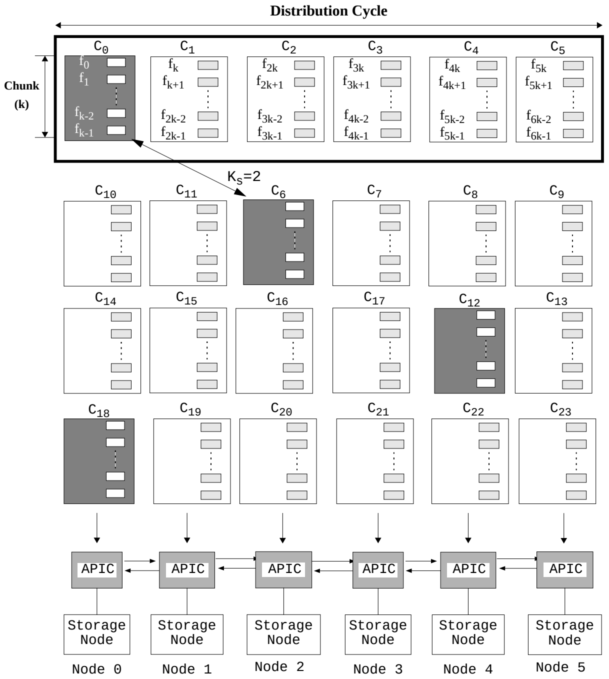

Figure 3: A Generalized Staggered Distributed Data Layouts

each containing D chunks for a layout on D storage nodes. The first chunk in a distribution cycle is called an **anchor** chunk and the node to which it is assigned is called the **anchor** node for that distribution cycle. As shown in the Figure 3, the anchor node for successive distribution cycles is staggered by a stagger factor k_s in a mod$D$ order. Clearly, changing k and k_s results in a new layout and thus, G-SDCL$_{k_s}(k)$ defines a family of data layouts. Note that in this scheme, the two consecutive chunks at the same node are separated in time by at least $(D - k_s)kT_f$ time units and at the most $(2D - k_s)kT_f$ time units. Thus, if the chunk is fetched as a single data unit, the stream is slowed down by at least $(D - k_s)$ from the perspective of each storage node or the throughput required per stream from each storage node is reduced by a factor of $(D - k_s)$. This in turns helps in masking the large prefetch latencies introduced by very slow storage devices at each node.

We will evaluate these distributed layouts using two performance metrics. The first one called *parallelism* (P_f), is defined as the **number of storage nodes participating concurrently in supplying the data for a document f.** The second metric called *concurrency* (C_f) defines the **number of active clients that can simultaneously access the same document f.** The value of P_f ranges from 1 to D, where D represents the number of storage nodes. P_f is D, when the data is distributed over all nodes, whereas, it is one when the entire document is confined to a single storage node. A higher value of P_f implies larger number of nodes are involved in transfer of data for each connection/request, which in turn improves node utilization and proportionately increases concurrency.

If each storage node n ($n \in [1 \ldots D]$) has an available sustained throughput of B_n, and the average storage/network throughput required for accessing the document f is R_f, then the concurrency supported by a layout with parallelism P_f is

$$C_f = \min_n \left(\frac{B_n P_f}{R_f} \right)$$

From above expression, we can see that the concurrency is a function of the parallelism supported by the data layout. Higher concurrency is desirable as it allows larger number of clients to simultaneously access the same document and thus minimizes the need for replicating the document to increase concurrent accesses.

2.2. Load balance properties of GSDCL

Consider a simple DCL$_1$ layout (recall chunk size $k = 1$) described in Section 2. When a document is accessed in a normal playout mode, the frames are retrieved and transmitted in a linear $(\bmod D)$ order. Thus, for a set S_f of any consecutive D frames (called "frame set"), the set of nodes S_n (called "node set") from which these frames are retrieved contains each node only once. Such a node set that maximizes parallelism is called a balanced node set. A balanced node set indicates that the load on each node, measured in number of frames, is uniform. However, when the document is accessed in an interactive mode, such as *ff* or *rw*, the load-balance condition may be violated. We define the fast forward (rewind) distance d_f (d_r) as the number of frames skipped in a fast forward (rewind) frame sequence. Consider a connection in a system with $D = 6$ storage nodes, a DCL$_1$ layout, and a fast forward implementation by skipping alternate frames. The frame sequence for normal playout is $\{0, 1, 2, 3, 4, 5, \ldots\}$, whereas for the fast forward the same sequence is altered to $\{0, 2, 4, 6, 8, 10, \ldots\}$. This implies that in this example, the odd-numbered nodes are never visited for frame retrieval during *ff*. Thus, when a connection is being serviced in *ff* mode, the load measured in terms of the number of frames retrieved doubles for even numbered nodes and reduces to zero for odd numbered nodes. In other words, the parallelism P_f is reduced from D to $D/2$ during *ff* of a connection and the concurrency must be proportionately reduced. Clearly, in presence of a large number of connections independently exhibiting interactivity, this can lead to occasional severe load imbalance in the system and can make it difficult to satisfy the QOS contract agreed upon with each client at the time of connection setup. Thus, if we can ensure that P_f and consequently, C_f are unaffected during *ff* or *rw*, we can guarantee load-balance situations. One way to do this, is to use only those frame skipping distances that do not affect P_f and C_f. We call such skipping distances as *"safe skipping distances"* (SSD). Thus, given a data layout, we want to know in advance all the SSDs a distributed data layout can support.

In the next section, we will characterize the safe skipping distances for various data layouts in the G-SDCL$_{k_s}(k)$ family of layouts. We will assume that the chunk size $k = 1$, and thus, call anchor chunk as anchor frame. Later, we will discuss implications of using non-unit k.

3. Safe Skipping Distances for Various Layouts

In this section, we will first derive some basic equations and later use them to derive SSDs for various layouts.

3.1. Basic Equations

We will assume that there are in all D storage node and the stagger distance is k_s. Consider any arbitrary distribution cycle in this layout with the anchor frame assigned to node p $(0 \leq p \leq D-1)$. Figure 4 illustrates this cycle. The j^{th} frame in this distribution cycle is then defined as in Equation 1, where f is the anchor frame in this cycle.

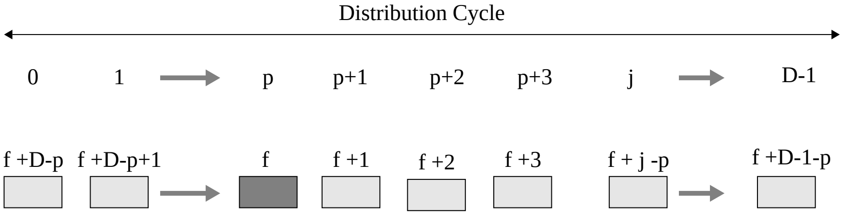

Figure 4: General distribution cycle with anchor node p

$$f_j = \begin{cases} f + D + j - p & 0 \leq j \leq p-1 \\ f + j - p & p \leq j \leq D-1 \end{cases} \tag{1}$$

The anchor frame in the i^{th} distribution cycle is iD and it is assigned to the node with id $ik_s \bmod D$. Thus, the j^{th} frame in the i^{th} distribution cycle of 0^{th} stagger cycle is given as per Equation 2

$$f_{ij} = \begin{cases} iD + D + j - ik_s \bmod D & 0 \leq j \leq iks \bmod D - 1 \\ iD + j - ik_s \bmod D & ik_s \bmod D \leq j \leq D-1 \end{cases} \tag{2}$$

l^{th} frame at node n:

Note that adding kD^2 to the frames in the 0^{th} stagger cycle, frames in any k^{th} stagger cycle can be computed. Also, the l^{th} frame at a node belongs to k_{cyc}^{th} stagger cycle where $k_{cyc} = l\ DIV\ D$ and k_{loc}^{th} distribution cycle where $k_{loc} = l \bmod D$.

Hence, the ID of the l^{th} frame at node n is given as

$$f^l = \begin{cases} k_{cyl}D^2 + k_{loc}D + D - (k_{loc}k_s \bmod D - n) & 0 \leq n \leq k_{loc}k_s \bmod D - 1 \ldots (a) \\ k_{cyl}D^2 + k_{loc}D + n - k_{loc}k_s \bmod D & k_{loc}k_s \bmod D \leq n \leq D - 1 \ldots (b) \end{cases} \tag{3}$$

Frame f to node n mapping:

The layout pattern repeats itself after every D cycles. This repeating pattern, called as *"Stagger Cycle"* contains D^2 frames. Hence a given frame with ID f will belong to a stagger cycle with ID

$$C_{stg} = f\ DIV\ D^2$$

The effective ID of the frame in this stagger cycle is given as:

$$f^{'} = f \bmod D^2$$

Each stagger cycle contains D distribution cycles. Therefore, the ID of the distribution cycle to which frame f belongs is $i = f^{'}\ DIV D$. Since, $f^{'}$ is less than D^2, $0 \leq i \leq D-1$. The stagger distance for this distribution cycle is $((ik_s) \bmod D$. The ID of the frame within the i^{th} distribution cycle is given as $f^{'} \bmod D$. Hence, the ID of the node to which the frame is assigned is as follows:

$$\begin{aligned} f^{'} &= f \bmod D^2 \\ f &\longmapsto \{n = [ik_s \bmod D + f^{'} \bmod D] \bmod D\} \end{aligned} \tag{4}$$

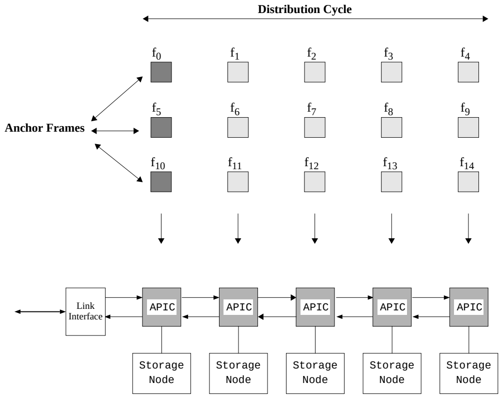

Figure 5: General distribution cycle with $k_s = 0$ and $k = 1$

3.2. GSDCL with $k_s = 0$: Distributed Cyclic Layout (DCL)

Let us take a closer look at the example G-SDCL shown in Figure 5. Assume that the fast forward starts from frame 0 with a fast forward distance d_f of 2. First $D = 5$ frames in the fast forward sequence are $\{0, 2, 4, 6, 8\}$, which are retrieved from a balanced node set $\{0, 2, 4, 1, 3\}$. If the fast forward distance is 3, the node set if altered to ordered set $\{0, 3, 1, 4, 2\}$, which is still balanced. It can be easily verified that the node set is balanced when $d_f = 4$, but is unbalanced when $d_f = D = 5$ or $d_f =$ integral multiple of D. Following theorem that relates D and d_f explicitly for this layout.

THEOREM 1. *Given a G-SDCL layout over D storage nodes with $k_s = 0$ and $k =$, the following holds true:Re*[1]

- *If the fast forward (rewind) distance d_f (d_r) is relatively prime to D, then*
  1. *The set of nodes S_n, from which consecutive D frames in fast forward (rewind) frame set S_f (S_r) are retrieved, is load-balanced.*
  2. *The fast forward (rewind) can start from any arbitrary frame (or node) number.*

Proof: We give a proof by **contradiction**. Let f be the number of the arbitrary frame from which the fast forward is started. The D frames in the fast forward frame set are then given as:

$$\{f, f + d_f, f + 2d_f, f + 3d_f, \ldots, f + id_f, \ldots f + jd_f + \ldots f + (D-1)d_f\}$$

Without any loss of generality, assume that two frames $f + id_f$ and $f + jd_f$, are mapped to the same node n_p.

Substitute $k_s = 0$ in Equation 3 (b). Clearly, the l^{th} frame at any node n in this layout is given as $f^l = n + lD$.Since any two frames mapped to the same node differ by an integral multiple of D, we have

$$(j - i) \quad = \quad k \times \frac{D}{d_f} \tag{5}$$

Two cases that arise are as follows:

[1] The result was first pointed out in a different form by Dr. Arif Merchant of the NEC Research Labs, Princeton, New Jersey, during the first author's summer research internship.

- **Case 1: k is not a multiple of d_f:** If D and d_f are relatively prime, then, $\frac{D}{d_f}$ cannot be an integer. However, $(j-i)$ is an integer. Thus, the Equation 5 cannot be true, which is a contradiction.
- **Case 2: k is a multiple of d_f:** If this condition is true, then $(j-i) = k_1 \times D$, where $k_1 = \frac{k}{d_f}$. However, this contradicts our assumption that the two selected frames are in the set which has only D frames and hence, can differ at the most $D-1$ in their ordinality.

Since the frame f from which fast-forward begins is selected arbitrarily, the claim 2 in the Theorem statement is also justified. The proof in the case of a rewind operation is similar and is not presented here. ∎

As per this theorem, if $D = 6$, skipping by all distances that are odd numbers $(1, 5, 7, 11\ldots)$ and are relatively prime to 6 will result in a balanced node set. We can see that if D is a prime number, then all distances d_f that are not multiples of D produce a balanced node set. Also, given a value D, there are always some distances d_f such as when d_f is a multiple of D or has a common factor with D, that cannot be safely supported.

3.3. GSDCL with $k_s = 1$ and $k = 1$

We will illustrate some of the special properties of this layout with an example. Let us consider example in Figure 6 and a *ff* implementation by skipping alternate frames (that is $d_f = 2$) starting from frame 0 . The original frame sequence $\{0, 1, 2, 3, 4, 5, 6, 7\}$ is then altered to $\{0, 2, 4, 6, 8, 10, 12, 14\}$. The node set for this new sequence is then altered from the balanced set $\{0, 1, 2, 3, 4, 5, 6, 7\}$ to $\{0, 2, 4, 6, 1, 3, 5, 7\}$. Clearly, this new node set is re-ordered but still is balanced. However, if $d_f = 3$, the similar node set is given as $\{0, 3, 6, 2, 5, 0, 4, 7\}$, which contains 0 twice and hence is unbalanced. It can be verified that cases $d_f = 4$, $d_f = 8$ and $d_f = m \times D$, where m is relatively prime to D produce balanced nodes sets as well. Note here that $2, 4$ are factors of $D = 8$, but 3 is not.

Theorem 2. *Given a* G-SDCL *layout with $k_s = 1$ over D storage nodes, and numbers $d_1, d_2, d_3, \cdots d_p$ that are factors of D, the following holds true:*

- **Load balance condition for fast forward:** *If the fast forward starts from an anchor frame f_a, with fast forward distance d_f, then the node set S_n is load-balanced, provided:*
  1. *$d_f = d_i$ (where $1 \leq i \leq p$) or*
  2. *$d_f = m \times D$ where m and D are relatively prime or*
  3. *$d_f = d_i + kD^2$ $(k > 0)$*
- **Load balance condition for rewind:** *The same result holds true for rewind if the rewind starts from a frame $2D-1$ after the anchor frame.*

Proof: We will prove this condition by **contradiction**. We will assume that there are maximum F_{max} frames in the stream. Let us first consider the case of fast forward.

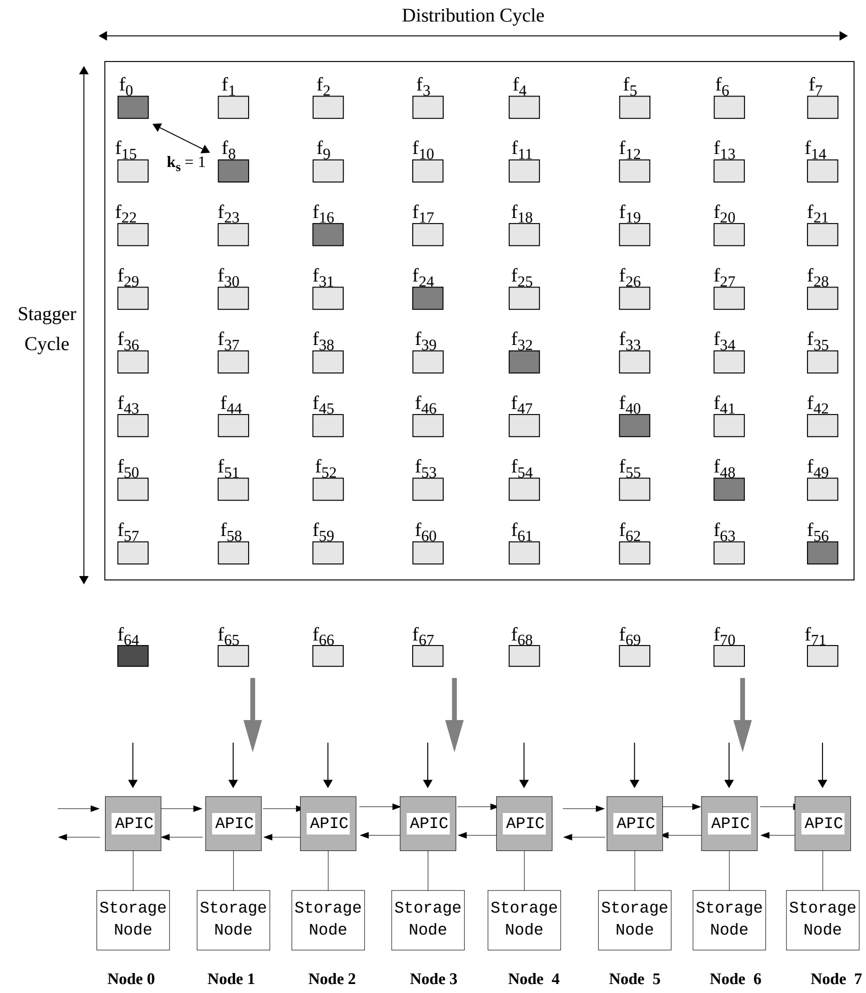
Distribution Cycle
Stagger Cycle
f_0 f_1 f_2 f_3 f_4 f_5 f_6 f_7
f_{15} $k_s = 1$ f_8 f_9 f_{10} f_{11} f_{12} f_{13} f_{14}
f_{22} f_{23} f_{16} f_{17} f_{18} f_{19} f_{20} f_{21}
f_{29} f_{30} f_{31} f_{24} f_{25} f_{26} f_{27} f_{28}
f_{36} f_{37} f_{38} f_{39} f_{32} f_{33} f_{34} f_{35}
f_{43} f_{44} f_{45} f_{46} f_{47} f_{40} f_{41} f_{42}
f_{50} f_{51} f_{52} f_{53} f_{54} f_{55} f_{48} f_{49}
f_{57} f_{58} f_{59} f_{60} f_{61} f_{62} f_{63} f_{56}
f_{64} f_{65} f_{66} f_{67} f_{68} f_{69} f_{70} f_{71}
APIC
Storage Node
Node 0
Node 1
Node 2
Node 3
Node 4
Node 5
Node 6
Node 7

Figure 6: Staggered Distributed Cyclic Layout(SDCL) with $k_s = 1$

Fast Forward Operation

The fast forward be started from the anchor frame f_a. The set of D frames in the fast forward frame set is then given as

$$S_f = \{f_a, f_a + d_f, f_a + 2d_f, f_a + 3d_f, \ldots, f_a + id_f, \ldots f_a + jd_f + \ldots f_a + (D-1)d_f\} \qquad (6)$$

If n_a $(0 \le n_a \le D-1)$ is the node to which the anchor frame f_a belongs, the IDs of the nodes to which the successive $D-1$ frames in the fast-forward sequence are mapped, can be easily written down.

In particular, consider frame $f_a + d_f$. If the G-SDCL$_0(1)$ had been followed, this frame would have been mapped to storage node $(n_a + d_f) \bmod D$. But due to the staggered layout, a stagger factor of d_f DIV D must be added. Since the storage node ID is always less than D, the effective ID is given by mod$D$ of this number. Therefore, $f_a + d_f$ is mapped to

$$f_a + d_f \mapsto ((n_a + d_f) \bmod D + d_f \;\; DIV \;\; D) \bmod D$$

In general, we can write down the following mapping relation

$$\begin{array}{rcl} f_a & \mapsto & n_a, \\ f_a + d_f & \mapsto & ((n_a + d_f) \bmod D + d_f \ DIV \ D) \bmod D, \\ & \vdots & \\ f_a + id_f & \mapsto & ((n_a + id_f) \bmod D + id_f \ DIV \ D) \bmod D, \\ & \vdots & \\ f_a + (D-1)d_f & \mapsto & ((n_a + (D-1)d_f) \bmod D + (D-1)d_f \ DIV \ D)x \bmod D, \end{array} \tag{7}$$

Now we must we prove that none of frames in S_f map to the same node. for all the three cases mentioned in the theorem.

Note that the l^{th} frame at any node n in this layout is computed by substituting $k_s = 1$ in Eqn 3.

$$f^l = \begin{cases} n + k_{cy}D^2 + k_{loc} \times (D-1) + D & \text{if } k_{loc} > n \\ n + k_{cy}D^2 + k_{loc} \times (D-1) & k_{loc} \leq n \end{cases} \tag{8}$$

Also, note that every $n^{th}, (n+D)^{th}, (n+2D)^{th} \ldots$ frames at node n are anchor frames.

FF-PART I: d_f is a factor of D: Let's first prove that none of the frames other than f_a in the fast forward set belong to the node n_a. Since $d_f = d_x$ for some x $(0 \leq x \leq p)$, $d_f < D$. Hence, $f_a + d_f(D-1) < f_a + (D-1)(D-1)$. Using the Equation 3, the set of $D-1$ frames following frame f_a at n_a is given as:

$$F_{n_a} = \{f_a + 2D - 1, f_a + 3D - 2, \ldots f_a + k(D-1) + D, \ldots f_a + (D-1)(D-1) + D\}. \tag{9}$$

Clearly, $f_a + id_f < f_a + (D-1)(D-1) + D$ for any value of $i \leq D-1$. Hence, if any of the frames in the fast forward set reappears at n_a, then it must belong to F_{n_a}. This implies that

$$\begin{array}{rcl} f_a + id_f & = & f_a + k(D-1) + D \\ i & = & \dfrac{k(D-1) + D}{d_f} \end{array} \tag{10}$$

Since d_f is a non-trivial (non-unit) factor of D, it can not be a factor of $D-1$. So for Equation 10 to be true, d_f must divide k. However, if $k/d_f \geq 1$, then $i > (D-1)$ which is not true. This implies that no frames in the fast forward set, other than f_a, can belong to the anchor node.

Therefore, if any two of the remaining $D-1$ frames in the fast forward set appear at the same node, that node must be different than the anchor node. Without loss of generality assume that frames $f_a + id_f$ and $f_a + jd_f$ are mapped to the same storage node, say $p(0 \leq p \leq D-1$ and $p \neq n_a)$. Since, none of these is an anchor frame, $i, j \neq 0$ and $(1 \leq i, j \leq D-1)$. Also, let $i > j$. Given this, $i - j$ can be at most $D-2$. We note, from Eqn 8 above, that the difference between frame numbers of any two frames at the same storage can be written down in general from as follows:

$$(f_a + id_f) - (f_a + jd_f) = k_1 D^2 + k_2(D-1) + k_3 D \tag{11}$$

In equation 11, k_1, k_2 and k_3 are integers and $0 \leq k_1 \leq (F_{max}\ DIV\ D^2)$, $0 \leq k_2 \leq D-1$, and $-1 \leq k_3 \leq 1$. Also, if $k_1 \neq 0$ and $k_2 \neq 0$, then $k_3 = -1, 0, 1$. If $k_1 = 0$, then the two frames belong to the same stagger cycle, thus, implying $k_2 \neq 0$, $k_3 = 0$ or 1. Also, if $k_2 = 0$, then $k_3 = 0$.

Case I: d_f is a factor of non-zero k_1 and k_2

$$i - j = \begin{cases} k_5 D^2 + k_6(D-1) + \frac{D}{d_f} & \text{if } k_3 = 1 \\ k_5 D^2 + k_6(D-1) & \text{if } k_3 = 0 \\ k_5 D^2 + k_6(D-1) - \frac{D}{d_f} & \text{if } k_3 = -1 \end{cases} \tag{12}$$

Clearly, since $k_5, k_6 > 1$, $i - j > (D-2)$, which is a contradiction. Hence, Equation 12 can not be true,

Case II: $k_1 = 0$ and d_f is a factor of non-zero k_2: In this case the difference $i - j$ is given as

$$i - j = \begin{cases} k_6(D-1) + \frac{D}{d_f} & \text{if } k_3 = 1 \\ k_6(D-1) & \text{if } k_3 = 0 \end{cases} \tag{13}$$

Since d_f is a factor of D and $k_6 \geq 1$, Equation 13 implies $i - j > D - 2$, which is a contradiction.

FF-PART II: $d_f = m \times D$, where m is non-zero and relatively prime with D: Assume that the first frame in fast forward set in Equation 6 maps to n_a. Consider, first the $D-1$ frames from f_a as given in Equation 9. Clearly, if any frame in fast forward set reappears at n_a and belongs to this set, then

$$\begin{aligned} f_a + imD &= f_a + k(D-1) + D \\ i &= \frac{1}{m} + \frac{k(D-1)}{mD} \end{aligned} \tag{14}$$

However, $\frac{1}{m}$ is not an integer and hence equation 14 can not be true. Successive set of D frames at n_a are obtained by adding integral multiple of D^2 to the frames in the set $F_{n_a} \bigcup f_a$. If any frame $f_a + imD$ from F_{ff} reappears at n_a, then

$$\begin{aligned} f_a + imD &= f_a + k(D-1) + D + lD^2 \\ i &= \frac{k(D-1)}{mD} + \frac{1}{m} + \frac{l}{m}D \end{aligned}$$

or

$$\begin{aligned} f_a + imD &= f_a + lD^2 \\ i &= \frac{lD}{m} \end{aligned} \tag{15}$$

Clearly, equation 15 can not be true for reason similar to one mentioned for Equation 14. Since, m and D are relatively prime, Equation 15 can not be true. Assume that two frames $f_a + id_f$ and

$f_a + jd_f$ from S_f appear at the same node, say p ($p \neq n_a$, $0 \leq p \leq D-1$). Then, we can write down Equation 16.

$$\begin{aligned} i - j &= \frac{k_1 D^2 + k_2(D-1) + k_3 D}{mD} \\ &= \frac{k_1}{m} D + \frac{k_2}{m} \frac{D-1}{D} + \frac{k_3}{m} \end{aligned} \tag{16}$$

- **Case I: m is a factor of non-zero k_1, k_2:** In this case the difference $i - j$ is given as

$$i - j = \begin{cases} k_5 D + k_6 \frac{(D-1)}{D} + \frac{1}{m} & \text{if } k_3 = 1 \ldots \text{(a)} \\ k_5 D + k_6 \frac{(D-1)}{D} & \text{if } k_3 = 0 \ldots \text{(b)} \\ k_5 D + k_6 \frac{(D-1)}{D} - \frac{1}{m} & \text{if } k_3 = 0 \ldots \text{(c)} \end{cases} \tag{17}$$

  Clearly, part (a) and part (b) of Equation 17 can not be true. Since, $\frac{D-1}{D}$ is not an integer, part (b) of above equation also cannot be true.

- **Case II: $k_1 = 0$ and $k_2 \neq 0$:** Equation 16 reduces to the following equation,

$$i - j = \begin{cases} \frac{k_2(D-1)}{mD} + \frac{1}{m} & \text{if } k_3 = 1 \ldots (a) \\ \frac{k_2(D-1)}{mD} & \text{if } k_3 = 0 \ldots (b) \end{cases} \tag{18}$$

Clearly, Equation 18 (a) can not be true as $\frac{1}{m}$ is not an integer. The part (b) of the equation can not be true due to two reasons: one, even if m divides $D-1$, because $0 \leq k_2 \leq D-1$, $\frac{k_2}{D}$ is not an integer. Also, even if m divides k_2, $\frac{D-1}{D}$ is not an integer. Thus, equation 16 can never be true, which is a contradiction to our assumption. Hence, the proof.

FF-PART III: $d_f = d_i + kD^2$ ($k > 0$) The G-SDCL layout consists of a basic "stagger pattern" that repeats itself every D distribution cycles or D^2 frames. In other words, if we form sets of D cycles starting at an arbitrary distribution cycle, they all will have the same structure. If one set of D cycles is given, frames in the other set can be obtained by adding or subtracting integral multiple of D^2. Also, when skipping by distances d_i mentioned in PART-I, the frame set S_f will contain frames that belong to consecutive D distribution cycles. Combining these two facts we can easily see that $d_f = d_i + kD^2$ produce balanced node sets.

Thus, from **PART I, II, III**, we can see that $f_a + id_f$ and $f_a + jd_f$ can not belong to the same node. Since, we have D storage nodes and D frames in the fast forward set S_f and no two frames appear at the same node, each frame must be mapped to a separate node. This implies that, each node appears once in the node set S_n and is perfectly load-balanced.

Rewind Operation

Assume that the rewind operation starts from the frame $f_a + 2D - 1$, where f_a is an anchor frame that belongs to node n_a. The set of D frames in the rewind frame set is then given as

$$S_r = \{f_a{+}2D{-}1, f_a{+}d_r{+}2D{-}1 f_a{+}2d_f{+}2D{-}1, \ldots, f_a{+}id_f{+}2D{-}1, \ldots f_a{+}jd_f{+} \ldots f_a{+}(D{-}1)d_f{+}2D{-}1\} \tag{19}$$

RW-PART I: d_r is a factor of D: Let's first prove that none of the frames other than $f_a + 2D - 1$ in the rewind set belong to the node n_a. Since $d_r = d_x$ for some x $(0 \leq x \leq p)$, $d_r < D$. Hence, $f_a + 2D - 1 - d_f(D-1) > f_a + 2D - 1 + (D-1)(D-1)$. Using the Equation 3, the set of $D - 1$ frames before frame $f_a + 2D - 1$ at n_a is given as:

$$R_{n_a} = \{f_a, f_a - (D-1), f_a - 2(D-1), \ldots f_a - k(D-1), \ldots f_a - (D-1)(D-1)\}. \tag{20}$$

Clearly, $f_a + 2D - 1 - id_r > f_a - (D-1)(D-1)D$ for any value of $i \leq D-1$. Hence, if any of the frames in the fast forward set reappears at n_a, then it must belong to F_{n_a}. This implies that

$$\begin{aligned} f_a + 2D - 1 - id_r &= f_a - k(D-1)(0 \leq k \leq (D-1) \\ i &= \frac{k(D-1)}{d_r} + \frac{D-1}{d_r} + \frac{D}{d_r} \end{aligned} \tag{21}$$

Since d_r is a non-trivial (non-unit) factor of D, it can not be a factor of $D - 1$. Clearly, is Equationeq:RWFirstPart can not be true. This implies that no frames in the rewind set, other than f_a, can belong to the anchor node n_a.

Therefore, if any two of the remaining $D - 1$ frames in the rewind set S_r appear at the same node, that node must be different than the anchor node. Without loss of generality assume that frames $f_a - id_r$ and $f_a - jd_r$ are mapped to the same storage node, say $p(0 \leq p \leq D-1$ and $p \neq n_a)$. Since, none of these is an anchor frame, $i, j \neq 0$ and $(1 \leq i, j \leq D-1)$. Also, let $i > j$. Given this, $i - j$ can be at most $D - 2$. We note, from Eqn 8 above, that the difference between frame numbers of any two frames at the same storage can be written down in general from as follows:

$$(f_a + 2D - 1 - id_r) - (f_a + 2D - 1 - jd_r) \;=\; -(k_1 D^2 + k_2(D-1) + k_3 D) \tag{22}$$

In equation 22, k_1, k_2 and k_3 are integers and $0 \leq k_1 \leq (F_{max}\ DIV\ D^2)$, $0 \leq k_2 \leq D-1$, and $-1 \leq k_3 \leq 1$. Also, if $k_1 \neq 0$ and $k_2 \neq 0$, then $k_3 = -1, 0, 1$. If $k_1 = 0$, then the two frames belong to the same stagger cycle, thus, implying $k_2 \neq 0$, $k_3 = 0$ or 1. Also, if $k_2 = 0$, then $k_3 = 0$. For reasons similar to Part I in the proof for fast-forward case, Equation 22 can not be true, which is a contradiction.

RW-PART II: $d_r = m \times D$, where m is non-zero and relatively prime with D: Assume that the first frame in rewind frame set in Equation 19 maps to n_a. Consider, first the $D - 1$ frames from f_a as given in Equation 9. Clearly, if any frame in fast forward set reappears at n_a and belongs to this set, then

$$\begin{aligned} f_a + 2D - 1 - imD &= f_a - k(D-1)(0 \leq k \leq (D-1) \\ i &= \frac{k(D-1)}{mD} + \frac{D-1}{mD} + \frac{1}{m} \end{aligned} \tag{23}$$

However, $\frac{1}{m}$ is not an integer and hence equation 23 can not be true. Successive set of D frames at n_a are obtained by adding integral multiple of D^2 to the frames in the set $R_{n_a} \bigcup f_a$. If any frame $f_a + 2D - 1 - imD$ from S_r reappears at n_a, then

$$\begin{aligned} f_a + 2D - 1 - imD &= f_a - k(D-1) - lD^2 \\ i &= \frac{k(D-1)}{mD} + \frac{l}{m}D + \frac{2D-1}{mD} \end{aligned}$$

or

$$\begin{aligned} f_a + 2D - 1 - imD &= f_a - lD^2 \\ i &= \frac{lD}{m} + \frac{2D-1}{mD} \end{aligned} \tag{24}$$

Clearly, equation 24 can not be true for reason similar to one mentioned for Equation 23. Since, m and D are relatively prime, Equation 24 can not be true. Using arguments similar to one in Part II in case of fast-forward, it can be proved that any two frames $f_a + 2D - 1 - id_r$ and $f_a + 2D - 1 - jd_r$ from S_r will not appear at the same node p ($p \neq n_a$, $0 \leq p \leq D-1$).

RW-PART III: $d_f = d_i + kD^2$ $(k > 0)$ This can be easily seen using arguments similar to PART III. Thus, from **PART I, II, III**, we can see that $f_a - id_r$ and $f_a - jd_r$ can not belong to the same node. Since, we have D storage nodes and D frames in the rewind set S_r and no two frames appear at the same node, each frame must be mapped to a separate node. This implies that, each node appears once in the node set S_n and is perfectly load-balanced. Hence, the proof. ■

3.4. GSDCL with arbitrary $k_s > 0$

To study load balance properties of these generalized layouts, consider a G-SDCL with eight nodes ($D = 8$) and a stagger distance of 3 ($k_s = 3$) illustrated in Figure 7. Let the fast forward distance d_f be four frames ($d_f = 4$), which is a factor of $D = 8$. Assume that the fast forward starts from the frame $f = 8$. Then the fast forward frame set is $S_f = \{8, 12, 16, 20, 24, 28, 32, 36\}$. The set of nodes from which these frames are retrieved is given as $S_n = \{3, 7, 6, 2, 1, 5, 4, 0\}$ which is a balanced node set. On the contrary, a fast forward starting at the same frame with a distance $d_f = 3$ produces a node set $S_n = \{3, 6, 1, 7, 2, 5, 3, 6\}$ which is unbalanced. Similarly, it can be verified that $d_f = 2, 8$ produce balanced node sets, but $d_f = 5, 6, 7$ do not.

Now let us consider a G-SDCL with $k_s = 2$ shown in Figure 8. If the fast forward begins at frame $f = 16$ with a distance $d_f = 2$, the corresponding node set $\{4, 6, 0, 2, 6, 0, 2, 4\}$ is unbalanced. Similarly, distances $d_f = 3, 4, 5, 6, 7, 8$ produce unbalanced node sets. From these examples we conjecture that if the the stagger distance k_s is relatively prime with the number of nodes then a result similar to the one mentioned in Theorem 1 is possible. Specifically, we conjecture that the following result can be proved.

CONJECTURE 1. *Given a* G-SDCL *layout with a stagger distance k_s over D storage nodes and the numbers $d_1, d_2, d_3, \ldots d_p$ that the are factors of D, the following holds true.*

1. **Load balance condition for fast-forward:** *If the fast forward always starts from an anchor frame, with a fast forward distance d_f, the node mapping set S_n is load-balanced, provided:*

   (a) *k_s and D are relatively prime* **and**

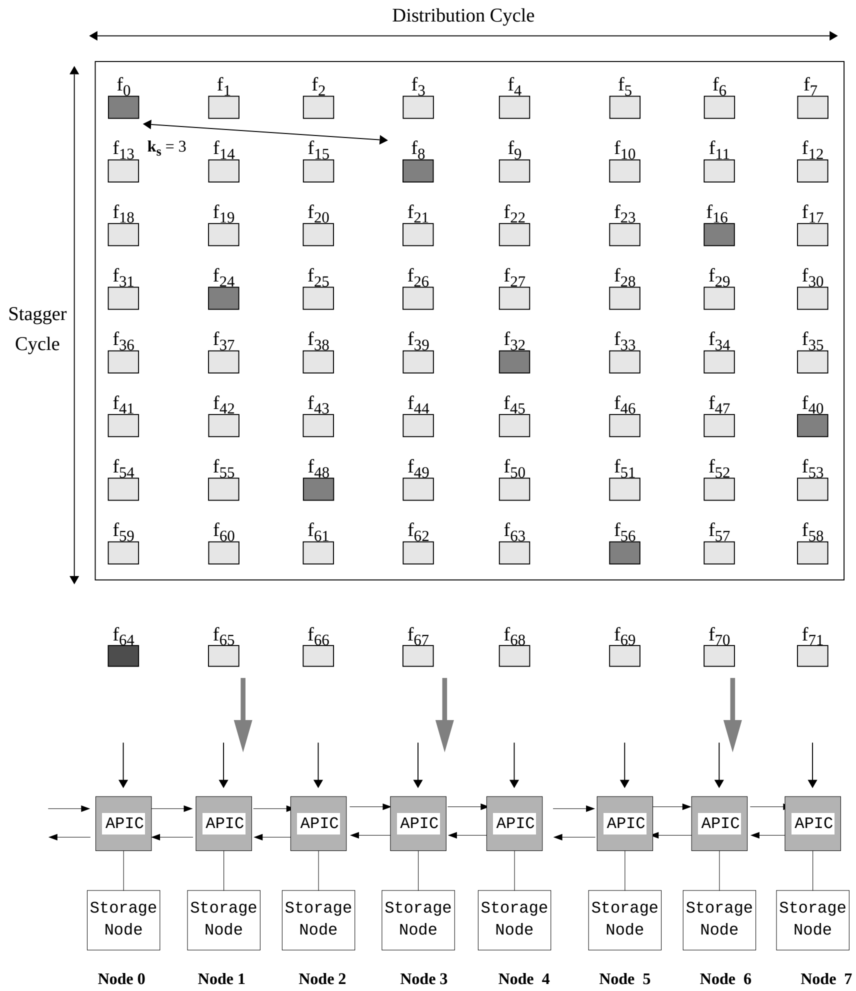
Distribution Cycle
Stagger Cycle
$k_s = 3$
f_0 f_1 f_2 f_3 f_4 f_5 f_6 f_7
f_{13} f_{14} f_{15} f_8 f_9 f_{10} f_{11} f_{12}
f_{18} f_{19} f_{20} f_{21} f_{22} f_{23} f_{16} f_{17}
f_{31} f_{24} f_{25} f_{26} f_{27} f_{28} f_{29} f_{30}
f_{36} f_{37} f_{38} f_{39} f_{32} f_{33} f_{34} f_{35}
f_{41} f_{42} f_{43} f_{44} f_{45} f_{46} f_{47} f_{40}
f_{54} f_{55} f_{48} f_{49} f_{50} f_{51} f_{52} f_{53}
f_{59} f_{60} f_{61} f_{62} f_{63} f_{56} f_{57} f_{58}
f_{64} f_{65} f_{66} f_{67} f_{68} f_{69} f_{70} f_{71}
APIC APIC APIC APIC APIC APIC APIC APIC
Storage Node Storage Node Storage Node Storage Node Storage Node Storage Node Storage Node Storage Node
Node 0 Node 1 Node 2 Node 3 Node 4 Node 5 Node 6 Node 7

Figure 7: Generalized Staggered Distributed Layout with $k_s = 3$

(b) $d_f = d_i$ $(1 \leq i \leq p-1)$ **or**

(c) $d_f = m \times D$, *where* m *and* D *are relatively prime* **or**

(d) $d_f = d_i + kD^2$ $(k > 0)$

2. **Load balance condition for rewind:** *A similar constraint defined in terms of anchor frame value,* k_s *and* D *holds for rewind operation.*

The proofs of this conjecture will be provided in a later version of this technical report.

4. Implications of Non-Unit Chunk Size

The discussion of load balance properties of data layouts in Section 2, 3 assumes that frame skipping required to implement *ff* and *rw* operations can be performed efficiently. This however may not be true.

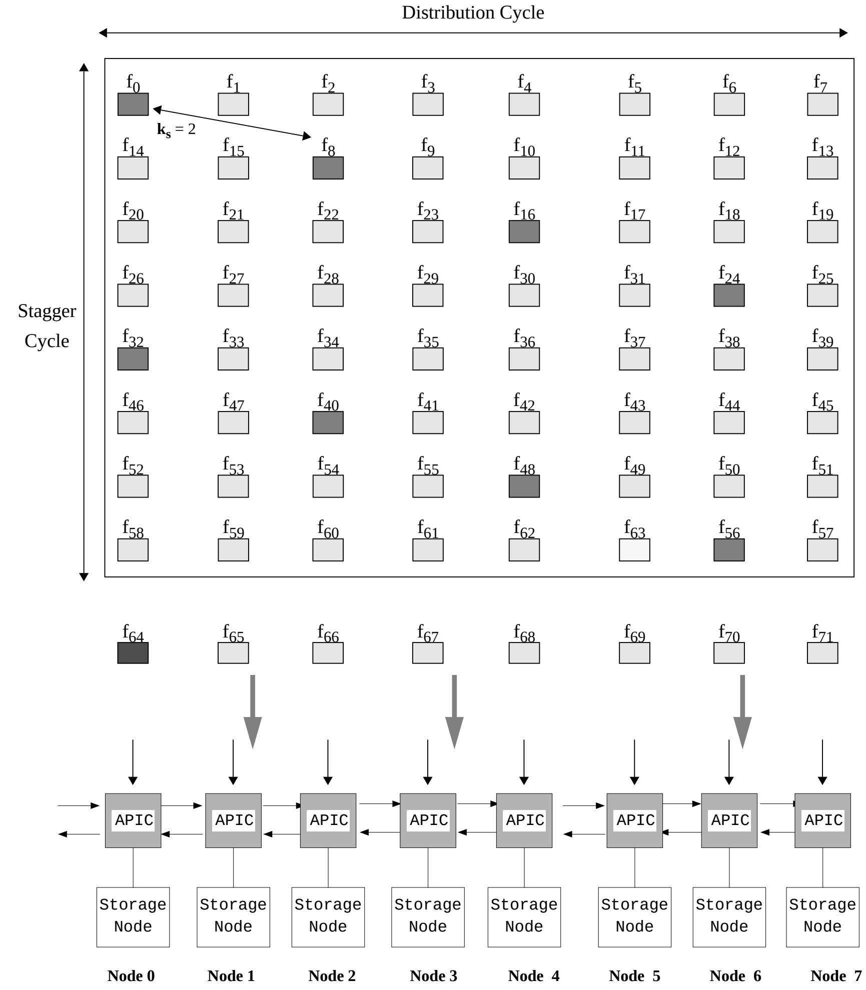
Distribution Cycle
Stagger Cycle
$k_s = 2$
f_0 f_1 f_2 f_3 f_4 f_5 f_6 f_7
f_{14} f_{15} f_8 f_9 f_{10} f_{11} f_{12} f_{13}
f_{20} f_{21} f_{22} f_{23} f_{16} f_{17} f_{18} f_{19}
f_{26} f_{27} f_{28} f_{29} f_{30} f_{31} f_{24} f_{25}
f_{32} f_{33} f_{34} f_{35} f_{36} f_{37} f_{38} f_{39}
f_{46} f_{47} f_{40} f_{41} f_{42} f_{43} f_{44} f_{45}
f_{52} f_{53} f_{54} f_{55} f_{48} f_{49} f_{50} f_{51}
f_{58} f_{59} f_{60} f_{61} f_{62} f_{63} f_{56} f_{57}
f_{64} f_{65} f_{66} f_{67} f_{68} f_{69} f_{70} f_{71}
APIC
APIC
APIC
APIC
APIC
APIC
APIC
APIC
Storage Node
Storage Node
Storage Node
Storage Node
Storage Node
Storage Node
Storage Node
Storage Node
Node 0
Node 1
Node 2
Node 3
Node 4
Node 5
Node 6
Node 7

Figure 8: Generalized Staggered Distributed Layout with $k_s = 2$

Suppose the chunk size used in $k = 1$. Several frames assigned to a node then can be saved contiguously on local storage device at a node retrieved in large chunks to minimize the seek and rotational latency by fetching large chunks in single seek operation. However, during the *ff* and *rw* implemented by frame skipping, individual frames must be read, which requires repositioning the disk head after every frame retrieval. For large skipping distances and small frames sizes that are common in compressed streams, each such read will suffer severe seek and rotation penalty. Such penalties can be minimized under heavy load, if prefetch load over multiple connection is randomly distributed over each disk and efficient disk scheduling algorithms such as those reported in [15] are used. However, under low or moderate loads, frame skipping may lead to poor disk utilization and cycle overflows.

Note that in case of $k > 1$, frame skipping causes load-imbalance. In other words, frame skipping is suitable for G-SDCL$_{k_s}(1)$ layouts.

An alternate approach to dealing with this problem is to always use a layout with a chunk size of k frames and implement *ff* and *rw* by increasing the granularity of skipping to chunks. This kind of chunk skipping is analogous to segment skipping discussed in [12]. It has the advantage that during normal playout as well as *ff* and *rw*, chunks are read from the disk in much the same way without any additional seek/rotation penalties. Also, all the results mentioned in Section 3 for frame skipping on G-SDCL$_{k_s}(k)$ layouts with any stagger distance apply to chunk skipping over G-SDCL$_{k_s}(k)$ layouts. However, the visual quality of such chunk skipping is likely to be unacceptable at large chunk sizes.

5. Related Work

This section briefly presents some of the related work. Keeton et. al. discuss schemes for placement of sub-band encoded video data in units of constant playout length on a two dimensional disk array [13]. They report simulation results which conclude that storage of multi-resolution video permits service to more concurrent clients than storage of single resolution video. Similarly, Zakhor et. al. report design of schemes for placing scalable sub-band encoded video data on a disk array. They focus only on the path from the disk devices to the memory and evaluate using simulation, layouts that use constant data or time length units. However, this paper does not address issues in the implementation of interactive operations.

Chen et. al. report data placement and retrieval schemes for an efficient implementation of *ff* and *rw* operations in a disk array based video server [12]. Our work is completely independent and concurrent this work and has similarities and differences [6, 7]. Chen et. al.'s paper assumes a disk array based small scale server whereas our work assumes a large scale server with multiple storage nodes, each with a disk array. They define a MPEG specific data layout unit called a segment, which is a collection of frames between two consecutive I frames. Our definition of a chunk is a data unit which requires a constant playout time at a given frames/sec rate. So a segment Chen et. al.'s segment is a special case of our chunk. Chen et. al. discuss two schemes for segment placement: in the first scheme, the segments are distributed on disks in a round robin fashion, in much the same way as our DCL$_k$ layout over multiple storage nodes. For *ff*/*rw* operation, they employ a segment selection method which ensures that over a set of retrieval cycles, each disk is visited only once. Thus, here the load balance is achieved over multiple retrieval cycles.In second segment placement scheme, the segments are placed on the disk array in such a way that for certain fast forward rates, the retrieval pattern for each round contains each disk only once. Our G-SDCL$_{k_s=1}(k)$ layout over storage nodes with stagger distance of one is similar to this second segment placement scheme. However, our result is more general, as it characterizes many more safe skipping rates for *ff* and gives a condition for safe implementation of rewind operation.

6. Conclusions

In this paper, we described distributed data layouts for clustered multimedia storage servers constructed out of distributed storage nodes interconnected high speed network. We illustrated a family of hierarchical, distributed layouts called *Generalized Staggered Distributed Data Layouts* (G-SDCL$_{k_s}$), that use constant time length logical units called chunks. For some of these layouts, we defined and proved a load-balance property that is required for efficient implementation of playout control operations such as fast-forward and rewind. These distributed data layouts and associated scheduling algorithms are currently being implemented in a testbed consisting of several Pentium PCs equipped with commercial disk arrays, and interconnected using an ATM switch. Experimental measurements will be used to evaluate our data layouts, scheduling schemes and and various node level layouts and scheduling options.

Acknowledgements

Some of the work reported in this paper was conducted during the first author's summer research internship at the Computer and Communications (C&C) Research Laboratory at NEC Research Institute, Princeton, New Jersey. We will therefore like to thank Dr. Kojiro Watanabe and Dr. Deepankar Raychaudhuri for this invaluable opportunity. We would also like to thank Dr. Arif Merchant and Daniel Reiniger, both at the NEC C&C Research Laboratory for many useful discussions.

References

[1] *MAGIC*$^{\text{TM}}$ *Media Server: A Scalable and Cost Effective Video Server*, Sarnoff Real Time Corporation, Princeton, NJ 08543.

[2] *Whittaker*$^{\text{TM}}$ *Media Server*, Whittaker Communication Corporation, Oregon.

[3] Hsieh, J., et al., "Performance of a Mass Storage System for Video-On-Demand," *Proceedings of IEEE INFOCOM'95*, pp. 771-778, April, 1995.

[4] Bernhardt, C., and Biersack, E., "A Scalable Video Server: Architecture, Design and Implementation," In *Proceedings of the Realtime Systems Conference*, pp. 63-72, Paris, France, Jan. 1995.

[5] Bernhardt, C., and Biersack, E., "The Server Array: A Scalable Video Server Architecture," To appear in *High-Speed Networks for Multimedia Applications*, editors, Danthine, A., Ferrari, D., Spaniol, O., and Effelsberg, W., Kluwer Academic Press, 1996.

[6] Buddhikot, M., Parulkar, G., M., and Cox, J., R., Jr., "Distributed Layout, Scheduling, and Playout Control in a Multimedia Storage Server," *Proceedings of the Sixth International Workshop on Packet Video*, Portland, Oregon, pp. C1.1 to C1.4, Sept. 26-27, 1994.

[7] Buddhikot, M., and Parulkar, G., M., "Distributed Scheduling, Data Layout and Playout Control in a Large Scale Multimedia Storage Server," Technical Report WUCS94-33, Department of Computer Science, Washington University in St. Louis, Sept. 1994.

[8] Buddhikot, M., Parulkar, G., and Cox, J., R., Jr., "Design of a Large Scale Multimedia Storage Server," *Journal of Computer Networks and ISDN Systems, pp. 504-517, Dec. 1994.*

[9] Buddhikot, M., and Parulkar, G., M., "Efficient Data Layout, Scheduling and Playout Control in MARS," Invited for publication in the *Special Issue of ACM/Springer Multimedia Systems Journal.*

[10] Chang, Ed, and Zakhor, A., "Scalable Video Placement on Parallel Disk Arrays," Image and Video Databases II, IS&T/SPIE International Symposium on Electronic Imaging: Science and Technology, San Jose, Feb. 1994.

[11] Chang, Ed, and Zakhor, A., "Variable Bit Rate MPEG Video Storage on Parallel Disk Arrays," First International Workshop on Community Networking, San Francisco, July 1994.

[12] Chen, M., Kandlur, D., and Yu, S., P., "Support for Fully Interactive Playout in a Disk-Array-Based Video Server," *Proceedings of Second International Conference on Multimedia, ACM Multimedia'94*, 1994.

[13] Keeton, K., and Katz, R., "The Evaluation of Video Layout Strategies on a High Bandwidth File Server," *Proceedings of International Workshop on Network and Operating Support for Digital Audio and Video (NOSSDAV'93)*, Lancaster, U. K., Nov., 1993.

[14] Tewari, R., Mukherjee, R., Dias, D., and and Harrick M. Vin. "Real-time Issues for Clustered Multimedia Servers," Submitted for publication, April 1995. (Also available as IBM Research Report RC 20020 (88561)).

[15] Vin, H., et al., "An Observation-Based Admission Control Algorithm for Multimedia Servers," *Proceedings of the IEEE International Conference on Multimedia Computing Systems (ICMCS'94)*, Boston, pp. 234-243, May 1994.