Application Data Sheet 37 CFR 1.76	Attorney Docket Number	92015039US01
	Application Number	

Title of Invention	SYSTEMS AND METHODS TO GENERATE A MISS RATIO CURVE FOR A CACHE WITH VARIABLE-SIZED DATA BLOCKS

The application data sheet is part of the provisional or nonprovisional application for which it is being submitted. The following form contains the bibliographic data arranged in a format specified by the United States Patent and Trademark Office as outlined in 37 CFR 1.76.
This document may be completed electronically and submitted to the Office in electronic format using the Electronic Filing System (EFS) or the document may be printed and included in a paper filed application.

Secrecy Order 37 CFR 5.2:

☐ Portions or all of the application associated with this Application Data Sheet may fall under a Secrecy Order pursuant to 37 CFR 5.2 (Paper filers only. Applications that fall under Secrecy Order may not be filed electronically.)

Inventor Information:

Inventor 1 — Remove

Legal Name

Prefix	Given Name	Middle Name	Family Name	Suffix
	Sari		SULTAN	

Residence Information (Select One) US Residency ● Non US Residency Active US Military Service

City	Toronto	Country of Residence i	CA

Mailing Address of Inventor:

Address 1	c/o Banting Institute, Heritage Building		
Address 2	100 College Street, Suite 413		
City	Toronto	**State/Province**	ON
Postal Code	M5G 1L5	**Country** i	CA

Inventor 2 — Remove

Legal Name

Prefix	Given Name	Middle Name	Family Name	Suffix
	Kia		SHAKIBA	

Residence Information (Select One) US Residency ◉ Non US Residency Active US Military Service

City	Toronto	Country of Residence i	CA

Mailing Address of Inventor:

Address 1	c/o Banting Institute, Heritage Building		
Address 2	100 College Street, Suite 413		
City	Toronto	**State/Province**	ON
Postal Code	M5G 1L5	**Country** i	CA

Inventor 3 — Remove

Legal Name

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	92015039US01
	Application Number	
Title of Invention	SYSTEMS AND METHODS TO GENERATE A MISS RATIO CURVE FOR A CACHE WITH VARIABLE-SIZED DATA BLOCKS	

Prefix	Given Name	Middle Name	Family Name	Suffix
	Albert		LEE	

Residence Information (Select One) US Residency (●) Non US Residency Active US Military Service

City	Toronto	Country of Residence i	CA

Mailing Address of Inventor:

Address 1	c/o Banting Institute, Heritage Building		
Address 2	100 College Street, Suite 413		
City	Toronto	State/Province	ON
Postal Code	M5G 1L5	Country i	CA

Inventor 4 Remove
Legal Name

Prefix	Given Name	Middle Name	Family Name	Suffix
	Michael		STUMM	

Residence Information (Select One) US Residency (●) Non US Residency Active US Military Service

City	Toronto	Country of Residence i	CA

Mailing Address of Inventor:

Address 1	c/o Banting Institute, Heritage Building		
Address 2	100 College Street, Suite 413		
City	Toronto	State/Province	ON
Postal Code	M5G 1L5	Country i	CA

Inventor 5 Remove
Legal Name

Prefix	Given Name	Middle Name	Family Name	Suffix
	Ming		CHEN	

Residence Information (Select One) US Residency (●) Non US Residency Active US Military Service

City	Kanata	Country of Residence i	CA

Mailing Address of Inventor:

Address 1	c/o 303 Terry Fox Drive, Suite 400		
Address 2			
City	Kanata	State/Province	ON
Postal Code	K2K 3J1	Country i	CA

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	92015039US01
	Application Number	

Title of Invention	SYSTEMS AND METHODS TO GENERATE A MISS RATIO CURVE FOR A CACHE WITH VARIABLE-SIZED DATA BLOCKS

Inventor 6 Remove

Legal Name

Prefix	**Given Name**	**Middle Name**	**Family Name**	**Suffix**
	Chung-Man	Abelard	CHOW	

Residence Information (Select One) US Residency (●) Non US Residency Active US Military Service

City	Kanata	Country of Residence i	CA

Mailing Address of Inventor:

Address 1	c/o 303 Terry Fox Drive, Suite 400		
Address 2			
City	Kanata	**State/Province**	ON
Postal Code	K2K 3J1	**Country** i	CA

All Inventors Must Be Listed - Additional Inventor Information blocks may be generated within this form by selecting the **Add** button. Add

Correspondence Information:

Enter either Customer Number or complete the Correspondence Information section below. For further information see 37 CFR 1.33(a).

☐ **An Address is being provided for the correspondence Information of this application.**

Customer Number	139050		
Email Address	cnaugler@mbm.com	Add Email	Remove Email
Email Address	aipatent@huawei.com	Add Email	Remove Email

Application Information:

Title of the Invention	SYSTEMS AND METHODS TO GENERATE A MISS RATIO CURVE FOR A CACHE WITH VARIABLE-SIZED DATA BLOCKS		
Attorney Docket Number	92015039US01	**Small Entity Status Claimed**	☐
Application Type	Nonprovisional		
Subject Matter	Utility		
Total Number of Drawing Sheets (if any)	9	**Suggested Figure for Publication (if any)**	

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	92015039US01
	Application Number	

Title of Invention	SYSTEMS AND METHODS TO GENERATE A MISS RATIO CURVE FOR A CACHE WITH VARIABLE-SIZED DATA BLOCKS

Filing By Reference:

Only complete this section when filing an application by reference under 35 U.S.C. 111(c) and 37 CFR 1.57(a). Do not complete this section if application papers including a specification and any drawings are being filed. Any domestic benefit or foreign priority information must be provided in the appropriate section(s) below (i.e., "Domestic Benefit/National Stage Information" and "Foreign Priority Information").

For the purposes of a filing date under 37 CFR 1.53(b), the description and any drawings of the present application are replaced by this reference to the previously filed application, subject to conditions and requirements of 37 CFR 1.57(a).

Application number of the previously filed application	Filing date (YYYY-MM-DD)	Intellectual Property Authority or Country

Publication Information:

- ☐ Request Early Publication (Fee required at time of Request 37 CFR 1.219)
- ☐ **Request Not to Publish.** I hereby request that the attached application not be published under 35 U.S.C. 122(b) and certify that the invention disclosed in the attached application **has not and will not** be the subject of an application filed in another country, or under a multilateral international agreement, that requires publication at eighteen months after filing.

Representative Information:

Representative information should be provided for all practitioners having a power of attorney in the application. Providing this information in the Application Data Sheet does not constitute a power of attorney in the application (see 37 CFR 1.32).
Either enter Customer Number or complete the Representative Name section below. If both sections are completed the customer Number will be used for the Representative Information during processing.

Please Select One:	● Customer Number	US Patent Practitioner	○ Limited Recognition (37 CFR 11.9)
Customer Number	139050		

Domestic Benefit/National Stage Information:

This section allows for the applicant to either claim benefit under 35 U.S.C. 119(e), 120, 121, 365(c), or 386(c) or indicate National Stage entry from a PCT application. Providing benefit claim information in the Application Data Sheet constitutes the specific reference required by 35 U.S.C. 119(e) or 120, and 37 CFR 1.78.
When referring to the current application, please leave the "Application Number" field blank.

Prior Application Status		Remove	
Application Number	Continuity Type	Prior Application Number	Filing or 371(c) Date (YYYY-MM-DD)

Additional Domestic Benefit/National Stage Data may be generated within this form by selecting the **Add** button. Add

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	92015039US01
	Application Number	
Title of Invention	SYSTEMS AND METHODS TO GENERATE A MISS RATIO CURVE FOR A CACHE WITH VARIABLE-SIZED DATA BLOCKS	

Foreign Priority Information:

This section allows for the applicant to claim priority to a foreign application. Providing this information in the application data sheet constitutes the claim for priority as required by 35 U.S.C. 119(b) and 37 CFR 1.55. When priority is claimed to a foreign application that is eligible for retrieval under the priority document exchange program (PDX)[i] the information will be used by the Office to automatically attempt retrieval pursuant to 37 CFR 1.55(i)(1) and (2). Under the PDX program, applicant bears the ultimate responsibility for ensuring that a copy of the foreign application is received by the Office from the participating foreign intellectual property office, or a certified copy of the foreign priority application is filed, within the time period specified in 37 CFR 1.55(g)(1).

Application Number	Country[i]	Filing Date (YYYY-MM-DD)	Remove Access Code[i] (if applicable)

Additional Foreign Priority Data may be generated within this form by selecting the **Add** button. Add

Statement under 37 CFR 1.55 or 1.78 for AIA (First Inventor to File) Transition Applications

☐ This application (1) claims priority to or the benefit of an application filed before March 16, 2013 and (2) also contains, or contained at any time, a claim to a claimed invention that has an effective filing date on or after March 16, 2013.
NOTE: By providing this statement under 37 CFR 1.55 or 1.78, this application, with a filing date on or after March 16, 2013, will be examined under the first inventor to file provisions of the AIA.

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	92015039US01
	Application Number	
Title of Invention	SYSTEMS AND METHODS TO GENERATE A MISS RATIO CURVE FOR A CACHE WITH VARIABLE-SIZED DATA BLOCKS	

Authorization or Opt-Out of Authorization to Permit Access:

When this Application Data Sheet is properly signed and filed with the application, applicant has provided written authority to permit a participating foreign intellectual property (IP) office access to the instant application-as-filed (see paragraph A in subsection 1 below) and the European Patent Office (EPO) access to any search results from the instant application (see paragraph B in subsection 1 below).

Should applicant choose not to provide an authorization identified in subsection 1 below, applicant **must opt-out** of the authorization by checking the corresponding box A or B or both in subsection 2 below.

NOTE: This section of the Application Data Sheet is **ONLY** reviewed and processed with the **INITIAL** filing of an application. After the initial filing of an application, an Application Data Sheet cannot be used to provide or rescind authorization for access by a foreign IP office(s). Instead, Form PTO/SB/39 or PTO/SB/69 must be used as appropriate.

1. Authorization to Permit Access by a Foreign Intellectual Property Office(s)

A. Priority Document Exchange (PDX) - Unless box A in subsection 2 (opt-out of authorization) is checked, the undersigned hereby **grants the USPTO authority** to provide the European Patent Office (EPO), the Japan Patent Office (JPO), the Korean Intellectual Property Office (KIPO), the State Intellectual Property Office of the People's Republic of China (SIPO), the World Intellectual Property Organization (WIPO), and any other foreign intellectual property office participating with the USPTO in a bilateral or multilateral priority document exchange agreement in which a foreign application claiming priority to the instant patent application is filed, access to: (1) the instant patent application-as-filed and its related bibliographic data, (2) any foreign or domestic application to which priority or benefit is claimed by the instant application and its related bibliographic data, and (3) the date of filing of this Authorization. See 37 CFR 1.14(h)(1).

B. Search Results from U.S. Application to EPO - Unless box B in subsection 2 (opt-out of authorization) is checked, the undersigned hereby **grants the USPTO authority** to provide the EPO access to the bibliographic data and search results from the instant patent application when a European patent application claiming priority to the instant patent application is filed. See 37 CFR 1.14(h)(2).

The applicant is reminded that the EPO's Rule 141(1) EPC (European Patent Convention) requires applicants to submit a copy of search results from the instant application without delay in a European patent application that claims priority to the instant application.

2. Opt-Out of Authorizations to Permit Access by a Foreign Intellectual Property Office(s)

☐ A. Applicant **DOES NOT** authorize the USPTO to permit a participating foreign IP office access to the instant application-as-filed. If this box is checked, the USPTO will not be providing a participating foreign IP office with any documents and information identified in subsection 1A above.

☐ B. Applicant **DOES NOT** authorize the USPTO to transmit to the EPO any search results from the instant patent application. If this box is checked, the USPTO will not be providing the EPO with search results from the instant application.

NOTE: Once the application has published or is otherwise publicly available, the USPTO may provide access to the application in accordance with 37 CFR 1.14.

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	92015039US01
	Application Number	
Title of Invention	SYSTEMS AND METHODS TO GENERATE A MISS RATIO CURVE FOR A CACHE WITH VARIABLE-SIZED DATA BLOCKS	

Applicant Information:

Providing assignment information in this section does not substitute for compliance with any requirement of part 3 of Title 37 of CFR to have an assignment recorded by the Office.

Applicant 1 Remove

If the applicant is the inventor (or the remaining joint inventor or inventors under 37 CFR 1.45), this section should not be completed. The information to be provided in this section is the name and address of the legal representative who is the applicant under 37 CFR 1.43; or the name and address of the assignee, person to whom the inventor is under an obligation to assign the invention, or person who otherwise shows sufficient proprietary interest in the matter who is the applicant under 37 CFR 1.46. If the applicant is an applicant under 37 CFR 1.46 (assignee, person to whom the inventor is obligated to assign, or person who otherwise shows sufficient proprietary interest) together with one or more joint inventors, then the joint inventor or inventors who are also the applicant should be identified in this section.

Clear

● Assignee	Legal Representative under 35 U.S.C. 117	Joint Inventor
Person to whom the inventor is obligated to assign.	Person who shows sufficient proprietary interest	

If applicant is the legal representative, indicate the authority to file the patent application, the inventor is:

Name of the Deceased or Legally Incapacitated Inventor:	
If the Applicant is an Organization check here.	☒
Organization Name	HUAWEI TECHNOLOGIES CANADA CO., LTD.

Mailing Address Information For Applicant:

Address 1	303 Terry Fox Drive		
Address 2	Suite 400		
City	Kanata	**State/Province**	ON
Country	CA	Postal Code	K2K 3J1
Phone Number		Fax Number	
Email Address			

Additional Applicant Data may be generated within this form by selecting the Add button. Add

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	92015039US01
	Application Number	

Title of Invention	SYSTEMS AND METHODS TO GENERATE A MISS RATIO CURVE FOR A CACHE WITH VARIABLE-SIZED DATA BLOCKS

Applicant 2 Remove

If the applicant is the inventor (or the remaining joint inventor or inventors under 37 CFR 1.45), this section should not be completed. The information to be provided in this section is the name and address of the legal representative who is the applicant under 37 CFR 1.43; or the name and address of the assignee, person to whom the inventor is under an obligation to assign the invention, or person who otherwise shows sufficient proprietary interest in the matter who is the applicant under 37 CFR 1.46. If the applicant is an applicant under 37 CFR 1.46 (assignee, person to whom the inventor is obligated to assign, or person who otherwise shows sufficient proprietary interest) together with one or more joint inventors, then the joint inventor or inventors who are also the applicant should be identified in this section.

Clear

◉ Assignee	Legal Representative under 35 U.S.C. 117	Joint Inventor

Person to whom the inventor is obligated to assign.	Person who shows sufficient proprietary interest

If applicant is the legal representative, indicate the authority to file the patent application, the inventor is:

Name of the Deceased or Legally Incapacitated Inventor:	

If the Applicant is an Organization check here. ☒

Organization Name	The Governing Council of the University of Toronto

Mailing Address Information For Applicant:

Address 1	Banting Institute, Heritage Building		
Address 2	100 College Street, Suite 413		
City	Toronto	**State/Province**	ON
Country	CA	Postal Code	M5G 1L5
Phone Number		Fax Number	
Email Address			

Additional Applicant Data may be generated within this form by selecting the Add button. Add

Assignee Information including Non-Applicant Assignee Information:

Providing assignment information in this section does not substitute for compliance with any requirement of part 3 of Title 37 of CFR to have an assignment recorded by the Office.

Assignee 1

Complete this section if assignee information, including non-applicant assignee information, is desired to be included on the patent application publication. An assignee-applicant identified in the "Applicant Information" section will appear on the patent application publication as an applicant. For an assignee-applicant, complete this section only if identification as an assignee is also desired on the patent application publication.

Remove

If the Assignee or Non-Applicant Assignee is an Organization check here. ☒

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	92015039US01
	Application Number	

Title of Invention	SYSTEMS AND METHODS TO GENERATE A MISS RATIO CURVE FOR A CACHE WITH VARIABLE-SIZED DATA BLOCKS

Organization Name	HUAWEI TECHNOLOGIES CANADA CO., LTD.

Mailing Address Information For Assignee including Non-Applicant Assignee:

Address 1	303 Terry Fox Drive		
Address 2	Suite 400		
City	Kanata	**State/Province**	ON
Country i	CA	Postal Code	K2K 3J1
Phone Number		Fax Number	
Email Address			

Additional Assignee or Non-Applicant Assignee Data may be generated within this form by selecting the Add button. Add

Assignee 2

Complete this section if assignee information, including non-applicant assignee information, is desired to be included on the patent application publication. An assignee-applicant identified in the "Applicant Information" section will appear on the patent application publication as an applicant. For an assignee-applicant, complete this section only if identification as an assignee is also desired on the patent application publication.

Remove

If the Assignee or Non-Applicant Assignee is an Organization check here. ☒

Organization Name	The Governing Council of the University of Toronto

Mailing Address Information For Assignee including Non-Applicant Assignee:

Address 1	Banting Institute, Heritage Building		
Address 2	100 College Street, Suite 413		
City	Toronto	**State/Province**	ON
Country i	CA	Postal Code	M5G 1L5
Phone Number		Fax Number	
Email Address			

Additional Assignee or Non-Applicant Assignee Data may be generated within this form by selecting the Add button. Add

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	92015039US01
	Application Number	
Title of Invention	SYSTEMS AND METHODS TO GENERATE A MISS RATIO CURVE FOR A CACHE WITH VARIABLE-SIZED DATA BLOCKS	

Signature:

Remove

NOTE: This Application Data Sheet must be signed in accordance with 37 CFR 1.33(b). **However, if this Application Data Sheet is submitted with the INITIAL filing of the application and either box A or B is not checked in subsection 2 of the "Authorization or Opt-Out of Authorization to Permit Access" section, then this form must also be signed in accordance with 37 CFR 1.14(c).**

This Application Data Sheet **must** be signed by a patent practitioner if one or more of the applicants is a **juristic entity** (e.g., corporation or association). If the applicant is two or more joint inventors, this form must be signed by a patent practitioner, **all** joint inventors who are the applicant, or one or more joint inventor-applicants who have been given power of attorney (e.g., see USPTO Form PTO/AIA/81) on behalf of **all** joint inventor-applicants.

See 37 CFR 1.4(d) for the manner of making signatures and certifications.

Signature	/Randall Marusyk/			Date (YYYY-MM-DD)	2022-11-07
First Name	Randall	Last Name	Marusyk	Registration Number	52423

Additional Signature may be generated within this form by selecting the Add button. Add

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1 The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these records.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3 A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent CooperationTreaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

SYSTEMS AND METHODS TO GENERATE A MISS RATIO CURVE FOR A CACHE WITH VARIABLE-SIZED DATA BLOCKS

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This is the first application filed for the present invention.

TECHNICAL FIELD OF THE INVENTION

[0002] This invention pertains generally to cache management and in particular, to methods and systems to allocate a cache using a miss ratio curve.

BACKGROUND

[0003] The miss rate or miss ratio of requests in a cache can be defined as the number of cache misses divided by the total number of cache memory requests over a given time interval:

$$miss\ ratio = \frac{[number\ of\ cache\ misses]}{[number\ of\ cache\ requests]}$$

[0004] Similarly, a miss ratio curve (MRC) for a cache presents a miss ratio as a function of a cache's size. In the prior art, one approach to generate a miss ratio curve is to calculate stack distances for data blocks (sequences of bits or bytes) in the cache and determine a frequency of occurrence for each stack distance, over a certain time, as a histogram. An MRC can be generated as an inverse cumulative distribution function (CDF) of the stack distance histogram, as presented in Mattson et al. (R. L. Mattson, J. Gecsei, D. R. Slutz and I. L. Traiger, "Evaluation techniques for storage hierarchies," *IBM Systems Journal*, vol. 9, no. 2, pp. 78-117, 1970). That method introduces the concept of a "stack distance", which refers to the position, in a stack of data blocks, of the most recently referenced data block. Although a basic stack distance computation method can be somewhat inefficient, other MRC methods based on stack distance can include further techniques to enhance its efficiency.

[0005] One limitation of an MRC technique according to prior art is the inability to support variable-sized data blocks.

[0006] Therefore, there is a need for MRC methods having the ability to support variable-sized data blocks, as well as systems to implement such methods, and such methods and systems would obviate or mitigate limitations of the prior art.

[0007] This background information is provided to reveal information believed by the applicant to be of possible relevance to the present invention. No admission is necessarily intended, nor should be construed, that any of the preceding information constitutes prior art against the present invention.

SUMMARY

[0008] Methods and systems disclosed allow the determination of a miss ratio curve (MRC) for caches using variable-sized memory blocks. Compatibility with variable-sized blocks allow an overall reduction in cache memory requirements and improved cache performance.

[0009] Technical benefits include compatibility with applications using data blocks having different sizes. For example, an application using two primary kinds of data, such as text messages and images, two different sizes of data blocks can be used such that one size is sufficient to accommodate a text message and the other size is sufficient to accommodate an image. Reserving the larger blocks sizes for images instead of text messages allows an economy of space. Methods according to embodiments allow the determination of a miss ratio curve, where a cache is using such variable block sizes. The miss ratio curve can then be used to allocate the cache size to be just sufficient to use of the application, and further spare memory space.

[0010] Embodiments of the present disclosure include a method to allocate a cache size comprising receiving a plurality of requests, each request for accessing a cached data block of a specified size; generating a stack of sets of counters, each set being a map of at least two counters; determining a plurality of stack distances from the stack of sets of counters; generating a miss ratio curve from a frequency distribution of the stack distances; determining the cache size according to the miss ratio curve; and allocating a cache of the determined cache size to satisfy a given performance requirement.

[0011] In an embodiment, each set of the sets of counters can include a counter for each data block of the specified size.

[0012] In an embodiment, generating a stack of sets of counters can comprise adding, at successive time steps, one set of counters for each time step; initializing to value "1" a counter in the one set of counters when a data block having a memory size corresponding to the counter is first accessed; initializing to value "0" a counter in the one set of counters when a data block having memory size other than the memory size corresponding to the counters is accessed; and incrementing existing counters when a data block having a memory size corresponding to the counter is first accessed.

[0013] In an embodiment, determining a stack distance from the stack of sets of counters, at a time step t, can comprise locating, at the time step t, a first counter $c_{i,t}^{block\ size}$ at row i and a second counter $c_{i+1,t}^{block\ size}$ at row i+1, each counter for the same block size, such that the counter $c_{i,t}^{block\ size}$ at row i does not increment and the counter $c_{i+1,t}^{block\ size}$ at row i+1 does increment; and selecting the value of the counter $c_{i,t}^{block\ size}$ that does not increment, wherein the block size is the block size accessed at time step t.

[0014] In an embodiment, each counter can be for counting unique accesses to cached data blocks having the same size as a data block being requested.

[0015] In an embodiment, a method can further comprise hashing, for each request, an address key of the requested data block into a binary sequence; and adding each resulting binary sequence to a corresponding counter of the set of counters.

[0016] In an embodiment, a method can further comprise merging a present counter with previous counters having the same memory size; computing a number of cache hits using a current stack of counters and the previous stack of counters; and considering the number of hits in the frequency distribution of the stack distances for generating the miss ratio curve.

[0017] In an embodiment, the size of a data block can be B^m bytes, B and m being integers, and a number n_c of counters in a set of counters can be limited to $n_c = m + 1$.

[0018] In an embodiment, a method can further comprise pruning a counter if the counter's value is at least $(1 - \delta)$ times the value of a corresponding counter in the next oldest set of counters, δ being a real number selected based on a memory limitation.

[0019] In an embodiment, a method can further comprise, if the number of counters is greater than a threshold determined by memory space limitations, invoking successive pruning operations, while increasing δ at each repetition, until at least one of the existing counters is pruned.

[0020] In an embodiment, a method can further comprise, if a request is for a memory block of a size that is not available, accessing an available memory block of a size corresponding to a rounding up of the memory block size being requested.

[0021] In an embodiment, a counter can be a hyperloglog (HLL) counter.

[0022] In an embodiment, generating a miss ratio curve from a frequency distribution of stack distances can comprise generating an inverse cumulative distribution function (CDF) of the frequency distribution of stack distances.

Embodiments of the present disclosure include a system for allocating a cache size comprising at least one processor, at least one cache, and machine readable memory storing machine readable instructions which when executed by the at least one processor, configures the at least one processor to receive a plurality of requests, each request for accessing a cached data block of a specified size; generate a stack of sets of counters, each set being a map of at least two counters; determine a plurality of stack distances from the stack of sets of counters; generate a miss ratio curve from a frequency distribution of the stack distances; determine the cache size according to the miss ratio curve; and allocate a cache of the determined cache size to satisfy a given performance requirement.

[0023] In an embodiment, each set of the sets of counters includes a counter for each data block of the specified size.

[0024] In an embodiment, each counter is for counting unique accesses to cached data blocks having the same size as a data block being requested.

[0025] In an embodiment, machine readable memory storing machine readable instructions can further configure the at least one processor to prune a counter if the counter's value is at least $(1 - \delta)$ times the value of a corresponding counter in the next oldest set of counters, δ being a real number selected based on a memory limitation.

[0026] In an embodiment, machine readable memory storing machine readable instructions can further configure the at least one processor to, if the number of counters is greater than a threshold determined by memory space limitations, invoke successive pruning operations, while increasing δ at each repetition, until at least one of the existing counters is pruned.

[0027] In an embodiment, machine readable memory storing machine readable instructions can further configure the at least one processor to access an available memory block of a size corresponding to a rounding up of the memory block size being requested.

[0028] In an embodiment, a counter is a hyperloglog (HLL) counter.

[0029] Embodiments have been described above in conjunction with aspects of the present invention upon which they can be implemented. Those skilled in the art will appreciate that embodiments may be implemented in conjunction with the aspect with which they are described but may also be implemented with other embodiments of that aspect. When embodiments are mutually exclusive, or are incompatible with each other, it will be apparent to those skilled in the art. Some embodiments may be described in relation to one aspect, but may also be applicable to other aspects, as will be apparent to those of skill in the art.

BRIEF DESCRIPTION OF THE DRAWINGS

[0030] Fig. 1a illustrates the state of a counter stack as it evolves in time.

[0031] Fig. 1b illustrates the counter stack of Fig. 1a, with emphasis on the latest state used to determine a stack distance.

[0032] Fig. 2 is a table representing the data structure of a stack of counter sets, and how the counters evolve in time, according to an embodiment.

[0033] Fig 3 is a schematically illustrates three main steps to generate an MRC, according to embodiment

[0034] Fig. 4 is a flowchart illustrating a method to process a batch of requests to access variable-sized blocks, before determining a stack distance according to embodiments

[0035] Fig. 5 is a flowchart illustrating steps for computing a number of hits in a cache based on a method according to an embodiment.

[0036] Fig. 6 illustrate three counter sets of 2 counters each, one counter for block size of 4K and one counter for a block size of 2K, according to an embodiment.

[0037] Fig. 7 is a graph comparing the mean absolute error between MRC calculations according to an embodiment, and MRC calculations according to prior art.

[0038] Fig. 8 is a graph comparing the throughput, in requests per second, for MRC calculations according to an embodiment, with the throughput for MRC calculations according to prior art.

[0039] Fig. 9 is a block diagram of an electronic device (ED) 952 illustrated within a computing and communications environment 950 that may be used for implementing the devices and methods disclosed herein, such as determining an MRC.

[0040] It will be noted that throughout the appended drawings, like features are identified by like reference numerals.

DETAILED DESCRIPTION

[0041] One limitation of an MRC technique according to prior art is the inability to support variable-sized data blocks, which is a feature in modern caching systems. For conventional MRC techniques, memory space is often partitioned into equally sized blocks and referred to as "pages". However, this can limit their use. The use of variable-sized blocks would be interesting for a communication system transmitting data blocks having various sizes, such as for example text messages and photos. With variable-size blocks, there is no need to use a photo-sized memory block (> 100 KB) for a text message if a text-sized memory block (< 1KB) is available. An MRC technique applicable to variable-sized blocks could overcome the limitation of having to use a large memory block (> 100 KB) for processing or transmitting a small (< 1KB) message.

[0042] Embodiments include methods to obtain a miss ratio curve (MRC) for caches supporting variable-sized memory blocks. Benefits of using variable-sized blocks include smaller memory requirements and improved performance of a cache. Therefore, if a miss ratio curve (MRC) can be obtained with variable-sized memory blocks, a caching system can be designed not only with variable sized memory blocks, but also with consideration of a corresponding MRC calculation.

[0043] To improve cache utilization, a method according to embodiments can be implemented by a caching system provided by a cloud-based platform. Based on an MRC result generated, the cache size of a workload can be adjusted. As a result, a customer can return unused cache resources to the cloud-based platform and therefore pay less for a service, while the platform can resell the unused resources for more revenue.

[0044] A method according to embodiments can support variable-sized blocks, by using a data structure based on counter stacks.

[0045] In embodiments, a "counter stack" (i.e. a stack of counters) generally refers to a data structure for which a stack distance can be calculated. A stack distance calculation based on counter stacks can be used to efficiently generate an MRC and allow further performance improvement techniques such as "streaming traces", "pruning", "down sampling", and the use of a "HyperLogLog" (HLL) method to estimate the number of unique elements (cardinality) in a data set.

[0046] In an embodiment, an HLL method is a probabilistic method for determining the cardinality of a set of data, i.e. the number of unique elements in the set. It is based on an assumption that a hash function generates equally distributed hashed values. By using such hashed values, the HLL method can approximate the number of unique elements in a data set.

[0047] Fig. 1a illustrates the state of a counter stack as it evolves in time. A stack 105 of three counters is shown in a column at the left of a matrix. Each row 110 of the matrix corresponds to a counter and shows the counter's state at successive time steps 120. Each column 115 of the matrix represents the state of the counter stack 105 at a certain time step 120. A counter's value is a matrix element the position of which can be identified by a counter number y and a state at time t: $c_{y,t}$.

[0048] A top row above the matrix illustrates a trace 125 of accesses to data blocks including data block a, data block b, and possibly more, by a cache client. The matrix is populated as follows. For counter c_1 between time step "1" and time step "3", two different elements are accessed: data block “a” and data block “b”. Therefore, the value for counter c_1 at time step t = 3 is $c_{1,3} = 2$. For counter c_2 between time step t = 1 and time step t = 2, only one unique element is accessed: data block "b”, and therefore the value for counter c_2 at time step t = 2 is $c_{2,2} = 1$. The value of a counter at time t, shown as a matrix element, is the number of different data blocks having been accessed between time step t = 1 and time step t.

[0049] Fig. 1b illustrates the counter stack of Fig. 1a, with emphasis on the latest state used to determine a stack distance. After a matrix is populated with counter values for accesses to “a” and/or “b” as in Fig. 1a, the method of Fig. 1b involves looking at the next-to-last and last columns (columns 2 and 3, or time steps 2 and 3), the last column being the "latest state" when a stack distance is to be determined. Therefore, each time a stack is updated with a new column in Fig 1a, a new stack distance can be determined as in Fig. 1b. There is one stack distance per counter, and between for example time 1 and time 2, the stack distance for a counter is the counter’s value. Determining a stack distance with a counter stack can involve identifying in the latest state (column) 130 a counter for which the value has not increased, for example:

- From time step 2 to time step 3, the state of counter c_1 has not increased from value 2 to value 2, therefore it is acceptable.
- From time step 2 to time step 3, the state of counter c_2 has not increased from value 1 to value 1, therefore it is acceptable.
- From time step 2 to time step 3, The state of counter c_3 has been initialized from no value to value 1, therefore it is not acceptable.

[0050] Then, from the acceptable counters c_1 and c_2, the method involves selecting from counter c_1 and c_2 the counter for which the next counter, c_2 and c_3 respectively, increases in value from time step 2 to time step 3. For example:

- For counter c_1, the next counter c_2 does not increase in value from 1 to 1, and therefore counter c_1 is not acceptable.
- For counter c_2, the next counter c_3 does increase from no value to 1, therefore counter c_2 is acceptable and because it is the only counter being acceptable for both criteria, it is selected.

- Identifying for the selected counter c_2 its row number, which is 2, and its column number, which is 3.

[0051] By selecting counter c_2, the above method has identified when the data block accessed at time step "3", i.e. data block "b", was last accessed, and the answer is time step "2", because counter c_2 does not increase in value from 1 to 1. The stack distance, between the access to data block b at time t = 3 and the last access to data block b at time = 2, is the value in counter $c_{2,3}$, which is "1".

[0052] In Fig. 1a and 1b, the data blocks (a, b) shown in the top row 125, and that are successively accessed, have similar or identical sizes. Embodiments, however, allow for data blocks with different sizes, i.e. variable-sized blocks. To support variable-sized blocks, a feature of a method according to embodiments is that instead of including a mere stack of counters 105, an embodiment includes a stack of "counter sets", where each counter set C_i (i = 1, 2, 3, …) is a map. Each map entry corresponds to a counter $c_i^{block\ size}$ or $c_i^{size\ in\ K}$ identified by a counter set i and a block size, such as a size in K (i.e. KB or kilobytes). For example, a stack S can include six counter sets c_i, where each counter set C_i includes one counter c_i^{4K} for blocks of size 4K and one counter c_i^{2K} for blocks of size 2K:

$$S = \{c_1, c_2, c_3, c_4, c_5, c_6\}$$

$$S = \{c_1^{4K}, c_1^{2K}, c_2^{4K}, c_2^{2K}, c_3^{4K}, c_3^{2K}, c_4^{4K}, c_4^{2K}, c_5^{4K}, c_5^{2K}, c_6^{4K}, c_6^{2K}\}$$

[0053] For variable-sized blocks according to embodiments, a stack distance can be calculated as a sum of weighted block sizes, where each weight (multiplication factor) is a counter $c_i^{size\ in\ K}$ of a set S, as follows:

$$stack\ distance = \sum_{c_i^{size\ in\ K} \in S} c_i^{size\ in\ K} \times (Size\ in\ K)$$

where S is a set of counters $c_i^{size\ in\ K}$ acting as weights (i.e. multiplication factors).

[0054] A method according to embodiments can support variable-sized blocks, by using a data structure based on counter stacks. Initially, a the method involves receiving a plurality of requests, each request for accessing a cached data block of a specified size. Then, a stack of sets of counters is generated, each set being a map of at least two counters. Then, a plurality of stack distances can be determined from the stack of sets of counters and a miss ratio curve can be generated from a frequency distribution of the stack distances. A cache size can be selected according to the miss ratio curve; and a cache can be allocated to the determined cache size, such as to satisfy a given performance requirement.

[0055] In an embodiment, each set of the sets of counters can include a counter for each data block of the specified size, and each counter can be a hyperloglog (HLL) counter.

[0056] When generating a stack of sets of counters, the method involves adding, at successive time step, one set of counters for each time step. It can initialize to value “1” a counter in the one set of counters when a data block having a memory size corresponding to the counter is first accessed and initialize to value “0” a counter in the one set of counters, when a data block having memory size other than the memory size corresponding to the counter is accessed. Existing counters are incremented when a data block having a memory size corresponding to the counters is first accessed.

[0057] The following is an example of how a data structure for a stack of counter sets according to embodiments can support blocks of various sizes, when calculating a stack distance.

[0058] Fig. 2 is a table representing the data structure of a stack of counter sets, with each counter set containing a map of counters, and how each counter evolves in time, according to an embodiment. A stack 205 includes six counter sets 210, each counter set 210 contains a map of two 215 counters, each counter corresponding to a block size, i.e. 4K or 2K. Each counter is for counting unique accesses to cached data blocks having the same size as a data block being requested, and each row represents the state of a counter at different time steps 220.

[0059] In a stack 205 of counter sets according to an embodiment, a counter from a counter stack 105 of the prior art is replaced with a set of counters 210 from a stack of counter sets. The first column of the table in Fig. 2 therefore includes six distinct counter sets: C_1 , C_2, C_3, C_4, C_5, C_6, each set being an array of two counters $c_i^{size\ in\ K}$ shown in the second column, each counter of a set corresponding to one of two block sizes. For example, set C_1 includes 215 counter c_1^{4K} and counter c_1^{2K}. Each counter is added at a certain time step 220 from time t = 1 to time t = 6. And when a counter is added, it is initialized 235 to 0 or 1, depending on whether the size of the accessed data block is 2K or 4K. For example, at time step 2, counter c_2^{4K} is added to set C_2, and it is initialized to 0, while counter c_2^{2K} is added to the same set C_2 but initialized to 1 instead. The initial value of initialization corresponds to the memory size of the accessed data block t (230), which is 2K.

[0060] At each time step, a data block shown in the top row is accessed. For example, at time step 1, a data block "A" 225 with memory size 4K is accessed, and this initializes counter c_1^{4K} to value "1", which is for a 4K data block. In Fig. 2, data blocks identified with an upper case letter (i.e. A, B, C) indicate a 4K block size, and data blocks identified with a lower case letter (i.e. t, u) indicate a 2K block size.

[0061] At time step 2, there is an access to data block "t" 230, which has a size 2K (2 kilobytes). This initializes counter c_1^{2K}, as well as a new counter c_2^{2K}, which are both for data blocks of size 2K. Counter c_1^{4K} is not incremented because it is not for a data block of size 2K, and counter c_2^{4K} is initialized to "0", because it is also not for a data block of size 2K. At subsequent time steps, further counters are similarly initialized and incremented, depending on the size of the data block accessed at that time step. Accordingly, the earliest counter set c_1 210 is at the top of the table and the latest counter set C_6 is at the bottom.

[0062] At a time step 1, there is an access to data block "A" 225, which has size 4K. A counter set C_1 is added and initialized, and a counter c_1^{4K} is added to counter set C_1. Starting from when counter set C_1 is initialized, counter c_1^{4K} counts the number of unique accesses to blocks of size 4K, between time step 1 and the present time step, as can be seen by following its row from time step 1 to time step 6. The present time step can be noted with a second subscript. If the present time step is "6", then counter c_1^{4K} can be noted as $c_{1,6}^{4K}$. At t = 6 for example, the value of $c_{1,6}^{4K}$ is the number of unique accesses to data blocks of 4K, between t = 1 and t = 6. From t = 1 to t = 6, accesses to 4K data blocks are (A, B, C, A). The unique accesses in this time frame are to data blocks A, B, and C, hence the value of $c_{1,6}^{4K}$ is "3".

[0063] At time step 1, a first distinct element of size 4K (data block A) 225 is accessed. Therefore, counter c_1^{4K} is initialized to value 1, while counter c_1^{2K} is initialized to value 0.

[0064] At time step 2, there is an access to data block "t" 230, which has a size 2K (2 kilobytes). A counter set C_2 is added and initialized 235, including a counter c_2^{4K} for blocks of size 4K and a counter c_2^{2K} for blocks of size 2K. Because the accessed block's size is 2K, counter c_2^{4K} is initialized to 0 and counter c_2^{2K} is initialized to 1.

[0065] In an example, the counters can evolve in time, and at time step 6, there can be an access to data block A. The following is an example of a stack distance determination by a processing system, where data blocks are accessed as shown in Fig. 2.

[0066] To determine a stack distance for a counter set according to an embodiment represented by Fig. 2, an initial step can be identifying the last access to data block A, by locating a pair of counters for the same block size, $c_i^{size\ in\ K}$ and $c_{i+1}^{size\ in\ K}$, such that: counter $c_i^{size\ in\ K}$ does not increment and subsequent counter $c_{i+1}^{size\ in\ K}$ does increment. For example, from time step 5 to time step 6, there is no incrementation 240 from counter $c_{1,5}^{4K}$, which is 3, to counter $c_{1,6}^{4K}$, which is also 3. However, there is an incrementation 245 from counter $c_{2,5}^{4K}$, which is 2, to counter $c_{2,6}^{4K}$, which is 3.

[0067] Then, a subsequent step can be identifying the matrix element having the row number of the non-incrementing 240 counter c_1^{4K}, and the present column: row 1 and column 6. The value of that counter is 3, which can be used to evaluate a stack distance in terms of memory size as follows:

stack distance at time “6” =

$$= [(\text{value of } c_{1,6}^{4K}) \text{ x } 4K] + [(\text{value of } c_{1,6}^{2K}) \text{ x } 2K]$$

stack distance at time “6” = [3 x 4K] + [2 x 2K]

stack distance at time “6” = [12K] + [4K]

stack distance at time “6” = 16K

[0068] In other words, determining a stack distance from the stack of sets of counters, at a time step t, can comprise locating, at the time step t, a first counter $c_{i,t}^{block\ size}$ at row i and a second counter $c_{i+1,t}^{block\ size}$ at row i+1, each counter for the same block size, such that the counter $c_{i,t}^{block\ size}$ at row i does not increment and the counter $c_{i+1,t}^{block\ size}$ at row i+1 does increment; and selecting the value of the counter $c_{i,t}^{block\ size}$ that does not increment, where the block size is the block size accessed at time step t.

[0069] In the prior art, a stack distance is a counter’s value, which is the unique access of a block between two time steps. This can be viewed as number of data blocks (each block having a size in KB) between the two accesses. But because embodiments involve variable sized blocks instead of same size blocks, the number of block is multiplied by the corresponding block size.

[0070] In the above, a method according to an embodiment allows the determination of a stack distance where variably sized blocks are taken into account. Once many stack distances are available, a stack distance distribution can be plotted as a histogram (a distribution of frequency for each stack distance), and an MRC can be generated on a scatter plot, as an inverse cumulative distribution function (CDF) of the stack distance histogram. In other words, generating a miss ratio curve from a frequency distribution of stack distances comprises generating an inverse cumulative distribution function (CDF) of the frequency distribution of stack distances.

[0071] Fig 3 is a flowchart illustrating three main steps to generate an MRC, according to embodiment. Initially, a stack distance is determined each time a block is requested 255. Then, the frequency of each stack distance is obtained, allowing for a frequency distribution 260, i.e. a histogram to be determined. An inverse cumulative distribution function (CDF) of the histogram results in a miss curve ratio or hit ratio curve 265.

[0072] Fig. 4 is a flowchart illustrating a method to process a batch of requests to access variable-sized blocks, before determining a stack distance according to embodiments. Requests come in as batches and the method of this flowchart determines how many requests should be included at each time step t. A counter is a HyperLogLog data structure, and the method involves hashing, for each request, an address key of the requested data block into a binary sequence; and adding each resulting binary sequence to a corresponding counter of the set of counters, i.e. the HyperLogLog data structure. When a down sampling rate threshold is reached, the count of requests is sufficient, and the counter set can be processed with the stack of counter sets.

[0073] Each access to a memory block can be referred to as a request, and initially, a processing system receives 305 a batch of requests. Then, it can process 310 a request in the batch and create 315 a counter in a current counter set, that corresponds to the block size of the memory block requested. The processing system then adds 320 a hashed value of a request to the corresponding counter of the current counter set. Each counter is a HyperLogLog data structure that uses the hashed value of a request to determine an approximated unique count of requests.

[0074] If 325 the number of processed requests is greater than or equal to the down sampling rate, then the counter set can be processed with the stack 330. Otherwise 335, if there are other requests, another request can be processed 310; otherwise 340, another batch can be received 305, until none are left 345, at which point the process can end.

[0075] Fig. 5 is a flowchart illustrating steps for computing a number of hits in a cache based on a method according to an embodiment. The computed number of hits can then be added to a frequency distribution of stack distances (histogram), that can be used to draw a scatter plot.

[0076] For each counter 405 of a given block size, in a counter set C_i, each previous counter set C_{i-1} 410 is examined for a counter for the same block size 415. If a counter for that block size exists in the previous counter set C_{i-1}, then the current counter can be merged 420 with previous counters for the same memory block size. If none exist, then the current counter can be linked 425 to the previous counter set C_{i-1} for that block size.

[0077] When all participating block sizes have been examined, this method can include computing 430 the count of counters and organizing them into a stack. Then, the number of cache hits can be computed 435 by using the current stack of counters and the previous stack of counters.

[0078] The method can include adding 440 the number of hits to a histogram and creating 445 a new counter set for processing a subsequent batch of requests.

[0079] In other words, the method involves merging counters with previous counters having the same memory block size; computing a number of cache hits using a current stack of counters and the previous stack of counters; and considering the number of hits in a frequency distribution of the stack distances for generating the miss ratio curve.

[0080] By computing the number of hits, an embodiment provides the ability to allocate a required cache size, which is important for an auto scaling feature of a caching service.

[0081] A method according to embodiments can allow a smaller amount of memory usage to be sufficient. Such smaller amount can be achieved with data structures having a smaller number of block sizes, a smaller number of counter sets, and a smaller number of counters in each counter set.

[0082] In embodiments, a block size that is a power of 2 can be sufficient, but a base other than two can also be used. Also, an access to a memory block can be rounded up to the next block size. For example, if a counter set supports memory blocks of 2K and 4K, and there is request for to a 3.5K block, it can be rounded up to access a 4K block. More generally, if a request is for a memory block of a size that is not available, the access can be for an available memory block of a size corresponding to a rounding up of the memory block size being requested.

[0083] In embodiments, a memory block can have a block size ranging from 1 byte to 2^m bytes, where m is an integer. Once a value for exponent m is selected, the number n_c of counters in a set of counters can be limited to $n_c = m + 1$. More generally, the size of a memory block can be B^m bytes, B and m being integers, and a number n_c of counters in a set of counters is limited to $n_c = m + 1$.

[0084] A method according to embodiments can support counter pruning, counter set pruning, as well as variations.

[0085] A counter can be pruned whenever its value is at least $(1 - \delta)$ times the corresponding counter in the next oldest counter set, where δ is a number indicating the difference between two counters, as defined by a user, below which a method considers them to be the same counter, and prunes one of them. In an example, $\delta = 0.01$. More generally pruning a counter can be performed if the counter's value is at least $(1 - \delta)$ times the value of a corresponding counter in the next oldest set of counters, δ being a real number selected based on a memory limitation.

[0086] A set of counters can be pruned (counter set pruning) when the value in each one of its counters is at least $(1 - \delta)$ times the value of a corresponding counter in the next oldest set of counters.

[0087] Fig. 6 illustrate 3 counter sets of 2 counters each, one counter for a block size of 4K and one counter for a block size of 2K, according to an embodiment. In Fig. 5, counter $c_2^{4K} = 99$ can be pruned 505, because its value is at least 1 – 0.01 = 0.99 times the value of the next oldest counter $c_1^{4K} = 100$. Similarly, counter set c_3 can be pruned 510, because the values of its counters are at least 1 – 0.01 = 0.99 times the values of the next oldest counter set.

[0088] Fig. 7 is a graph comparing the mean absolute error between MRC calculations according to an embodiment, and MRC calculations according to prior art. These are for publicly available trace collections, i.e. sequences of requests accessing data blocks, including one from the Cambridge Microsoft Research Lab (MSR) (Dushyanth Narayanan, Austin Donnelly, and Antony Rowstron. Write off-loading: Practical power management for enterprise storage. ACM

Trans. Storage, pages 10:1 - 10:23, 10 2008.), one from Twitter (Juncheng Yang, Yao Yue, and KV Rashmi. A large scale analysis of hundreds of in-memory cache clusters at Twitter. In Proc. 14th Symp. On Operating Systems Design and Implementation (OSDI'20), pages 191–208, 2020), one from the SEC (U.S. Securities and Exchange Commission (SEC), James Ryans. Using the EDGAR log file data set. Available at SSRN 2913612, 2017.), and one from Wikipedia (Guido Urdaneta, Guillaume Pierre, and Maarten Van Steen. Wikipedia workload analysis for decentralized hosting. Computer Networks, 53(11):1830 – 1845, 2009).

[0089] Overall, low fidelity (LF) and high-fidelity (HF) methods according to embodiments perform with a mean absolute error (MAE) that is less than that of prior art method A at LF and HF (Jake Wires, Stephen Ingram, Zachary Drudi, Nicholas JA Harvey, and Andrew Warfield. Characterizing storage workloads with counter stacks. In Proc. 11th Symp. on Operating Systems Design and Implementation (OSDI'14), pages 335–349, 2014.), and less than that of prior art method B at sampling rates R0.01 and R0.1 (] Carl A Waldspurger, Nohhyun Park, Alexander Garthwaite, and Irfan Ahmad. Efficient MRC construction with SHARDS. In Proc. 13th USENIX Conf. on File and Storage Technologies (FAST'15), pages 95–110, 2015.).

[0090] Fig. 8 is a graph comparing the throughput, in requests per second, for MRC calculations according to an embodiment, with the throughput for MRC calculations according to prior art. Prior art results are for method A as in Fig. 6, as well as for a prior art method C (Frank Olken. Efficient methods for calculating the success function of fixed-space replacement policies. Master's thesis, University of California, Berkeley), 1981.). Methods according to embodiments are shown to have a greater throughput than those of prior art method A and prior art method C.

[0091] A counter can be replaced with a pointer that points to the next oldest corresponding counter.

[0092] In a method according to an embodiment, if memory space limitations are exceeded, the space complexity, i.e. the amount of memory space required to complete an execution, can be held constant by invoking successive pruning operations, while incrementing δ at each repetition, until at least one of the existing counters is pruned. More generally, if the number of counters is greater than a threshold determined by memory space limitations, successive pruning operations can be invoked, while increasing δ at each repetition, until at least one of the existing counters is pruned.

[0093] A method according to embodiments can define a minimum number of counters required , by supporting a number n of counters equal to $n = 2(m - l + 1)$, where m and l are integers defining the range of block sizes that are supported: from 2^l bytes to 2^m bytes. This method is used to define a lower bound on the number of counters needed.

[0094] Embodiments include a more practical way to predict caching usage, which can benefit any system in terms of resource management. Because a caching pattern is a fundamental pattern in a computer system, any chip or application using a cache can benefit from improved caching.

[0095] Fig. 9 is a block diagram of an electronic device (ED) 952 illustrated within a computing and communications environment 950 that may be used for implementing the devices and methods disclosed herein, such as a system for determining a cache miss ratio curve. The electronic device 952 typically includes a processor 954, such as a central processing unit (CPU), and may further include specialized processors such as a field programmable gate array (FPGA) or other such processor, a memory 956, a network interface 958 and a bus 960 to connect the components of ED 952. ED 952 may optionally also include components such as a mass storage device 962, a video adapter 964, and an I/O interface 968 (shown in dashed lines).

[0096] The memory 956 may comprise any type of non-transitory system memory, readable by the processor 954, such as static random-access memory (SRAM), dynamic random-access memory (DRAM), synchronous DRAM (SDRAM), read-only memory (ROM), or a combination thereof. In an embodiment, the memory 956 may include more than one type of memory, such as ROM for use at boot-up, and DRAM for program and data storage for use while executing programs. The bus 960 may be one or more of any type of several bus architectures including a memory bus or memory controller, a peripheral bus, or a video bus.

[0097] The electronic device 952 may also include one or more network interfaces 958, which may include at least one of a wired network interface and a wireless network interface. A network interface 958 may include a wired network interface to connect to a network 974, and also may include a radio access network interface 972 for connecting to other devices over a radio link. The network interfaces 958 allow the electronic device 952 to communicate with remote entities such as those connected to network 974.

[0098] The mass storage 962 may comprise any type of non-transitory storage device configured to store data, programs, and other information and to make the data, programs, and other information accessible via the bus 960. The mass storage 962 may comprise, for example, one or more of a solid-state drive, hard disk drive, a magnetic disk drive, or an optical disk drive. In some embodiments, mass storage 962 may be remote to the electronic device 952 and accessible through use of a network interface such as interface 958. In the illustrated embodiment, mass storage 962 is distinct from memory 956 where it is included and may generally perform storage tasks compatible with higher latency but may generally provide lesser or no volatility. In some embodiments, mass storage 962 may be integrated with a heterogeneous memory 956.

[0099] In an embodiment, a system for determining a miss ratio curve can comprise at least one processor 954; a machine readable memory 956 storing machine readable instructions which when executed by the at least one processor 954, configures the at least one processor 954 to receive a plurality of requests, each request for accessing a cached data block of a specified size; generate a stack of sets of counters, each set being a map of at least two counters; determine a plurality of stack distances from the stack of sets of counters; generate a miss ratio curve from a frequency distribution of the stack distances; determine the cache size according to the miss ratio curve; and allocate a cache of the determined cache size to satisfy a given performance requirement. The network interface 974 and I/O interface 968 can also allow for storage and/or processing to occur externally.

[00100] In some embodiments, electronic device 952 may be a standalone device, while in other embodiments electronic device 952 may be resident within a data center. A data center, as will be understood in the art, is a collection of computing resources (typically in the form of servers) that can be used as a collective computing and storage resource. Within a data center, a plurality of servers can be connected together to provide a computing resource pool upon which virtualized entities can be instantiated. Data centers can be interconnected with each other to form networks consisting of pools computing and storage resources connected to each by connectivity resources. The connectivity resources may take the form of physical connections such as ethernet or optical communications links, and in some instances may include wireless communication channels as well. If two different data centers are connected by a plurality of different communication channels, the links can be combined together using any of a number of techniques including the formation of link aggregation groups (LAGs). It should be understood that any or all of the computing, storage, and connectivity resources (along with other resources within the network) can be divided between different sub-networks, in some cases in the form of a resource slice. If the resources across a number of connected data centers or other collection of nodes are sliced, different network slices can be created.

[00101] Allocating a cache size means configuring a cache to have a certain selected size, or a certain selected maximum size.

[00102] Although the present invention has been described with reference to specific features and embodiments thereof, it is evident that various modifications and combinations can be made thereto without departing from the invention. The specification and drawings are, accordingly, to be regarded simply as an illustration of the invention as defined by the appended claims, and are contemplated to cover any and all modifications, variations, combinations, or equivalents that fall within the scope of the present invention.

WHAT IS CLAIMED IS:

1. A method to allocate a cache comprising:

receiving a plurality of requests, each request for accessing a cached data block of a specified size;

generating a stack of sets of counters, each set being a map of at least two counters;

determining a plurality of stack distances from the stack of sets of counters;

generating a miss ratio curve from a frequency distribution of the stack distances;

determining the cache size according to the miss ratio curve; and

allocating a cache of the determined cache size to satisfy a given performance requirement.

2. The method of claim 1, wherein each set of the sets of counters includes a counter for each data block of the specified size.

3. The method of claim 1, wherein generating a stack of sets of counters comprises: adding, at successive time steps, one set of counters for each time step;

initializing to value "1" a counter in the one set of counters when a data block having a memory size corresponding to the counter is first accessed;

initializing to value "0" a counter in the one set of counters when a data block having memory size other than the memory size corresponding to the counters is accessed; and

incrementing existing counters when a data block having a memory size corresponding to the counter is first accessed.

4. The method of claim 3, wherein determining a stack distance from the stack of sets of counters, at a time step t, comprises locating, at the time step t, a first counter $c_{i,t}^{block\ size}$ at row i and a second counter $c_{i+1,t}^{block\ size}$ at row i+1, each counter for the same block size, such that the counter $c_{i,t}^{block\ size}$ at row i does not increment and the counter $c_{i+1,t}^{block\ size}$ at row i+1 does increment; and selecting the value of the counter $c_{i,t}^{block\ size}$ that does not increment, wherein the block size is the block size accessed at time step t.

5. The method of claim 1, wherein each counter is for counting unique accesses to cached data blocks having the same size as a data block being requested.

6. The method of claim 1, further comprising:

hashing, for each request, an address key of the requested data block into a binary sequence; and

adding each resulting binary sequence to a corresponding counter of the set of counters.

7. The method of claim 3, further comprising:

merging a present counter with previous counters having the same memory size;

computing a number of cache hits using a current stack of counters and the previous stack of counters; and

considering the number of hits in the frequency distribution of the stack distances.

8. The method of claim 1, wherein the size of a data block is B^m bytes, B and m being integers, and a number n_c of counters in a set of counters is limited to $n_c = m + 1$.

9. The method of claim 1, further comprising: pruning a counter if the counter's value is at least $(1 - \delta)$ times the value of a corresponding counter in the next oldest set of counters, δ being a real number selected based on a memory limitation.

10. The method of claim 9, further comprising: if the number of counters is greater than a threshold determined by memory space limitations, invoking successive pruning operations, while increasing δ at each repetition, until at least one of the existing counters is pruned.

11. The method of claim 1, further comprising:

if a request is for a memory block of a size that is not available,

accessing an available memory block of a size corresponding to a rounding up of the memory block size being requested.

12. The method of claim 1, wherein a counter is a hyperloglog (HLL) counter.

13. The method of claim 1, wherein
generating a miss ratio curve from the frequency distribution of stack distances comprises generating an inverse cumulative distribution function (CDF) of the frequency distribution of stack distances.

14. A system for allocating a cache size comprising:
at least one processor,
at least one cache, and
machine readable memory storing machine readable instructions which when executed by the at least one processor, configures the at least one processor to
receive a plurality of requests, each request for accessing a cached data block of a specified size;
generate a stack of sets of counters, each set being a map of at least two counters;
determine a plurality of stack distances from the stack of sets of counters;
generate a miss ratio curve from a frequency distribution of the stack distances;
determine the cache size according to the miss ratio curve; and
allocate a cache of the determined cache size to satisfy a given performance requirement.

15. The system of claim 14, wherein each set of the sets of counters includes a counter for each data block of the specified size.

16. The system of claim 14, wherein each counter is for counting unique accesses to cached data blocks having the same size as a data block being requested.

17. The system of claim 14, wherein the machine readable memory storing machine readable instructions further configures the at least one processor to prune a counter if the counter's value is at least $(1 - \delta)$ times the value of a corresponding counter in the next oldest set of counters, δ being a real number selected based on a memory limitation.

18. The system of claim 17, wherein the machine readable memory storing machine readable instructions further configures the at least one processor to, if the number of counters is greater than a threshold determined by memory space limitations, invoke successive pruning operations, while increasing δ at each repetition, until at least one of the existing counters is pruned.

19. The system of claim 14, wherein the machine readable memory storing machine readable instructions further configures the at least one processor to access an available memory block of a size corresponding to a rounding up of the memory block size being requested.

20. The system of claim 14, wherein a counter is a hyperloglog (HLL) counter.

ABSTRACT

A method to obtain a cache miss ratio curve where a memory blocks of a cache have variable block sizes. By stacking sets of counters, each set being for a different block size, a stack distance for variable block sizes can be obtained and used to determine a miss ratio curve. Such curve can then be used to select a cache size that is appropriate for an application without requiring excessive memory. Methods can be used for batches of request, can apply limits to block sizes, and rounding for intermediary block sizes, they can be used with pruning, and their space complexity can be held constant.

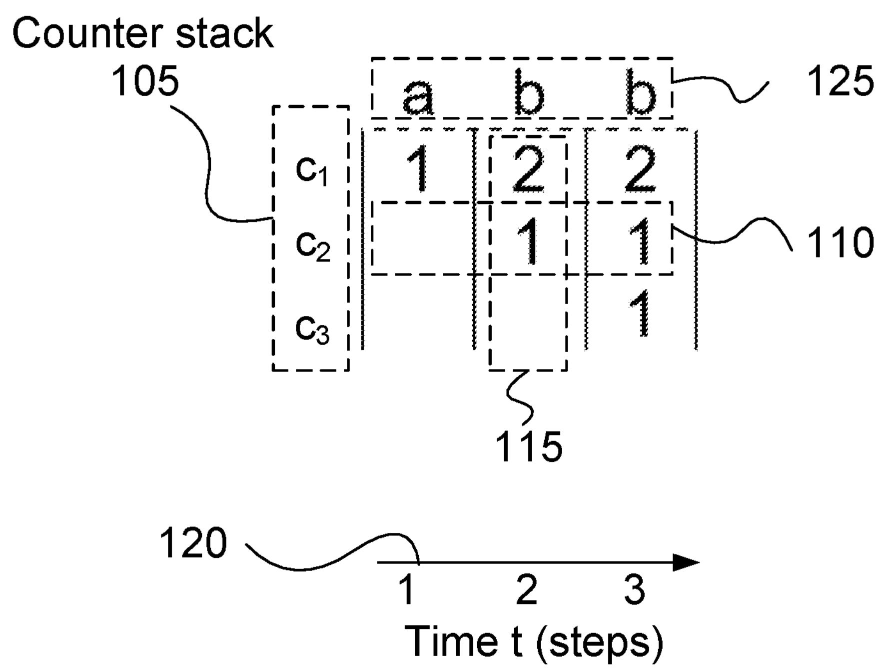

Counter stack
105
a b b
125
c1
c2
c3
1 2 2
1 1
1
110
115
120
1 2 3
Time t (steps)

Fig. 1a

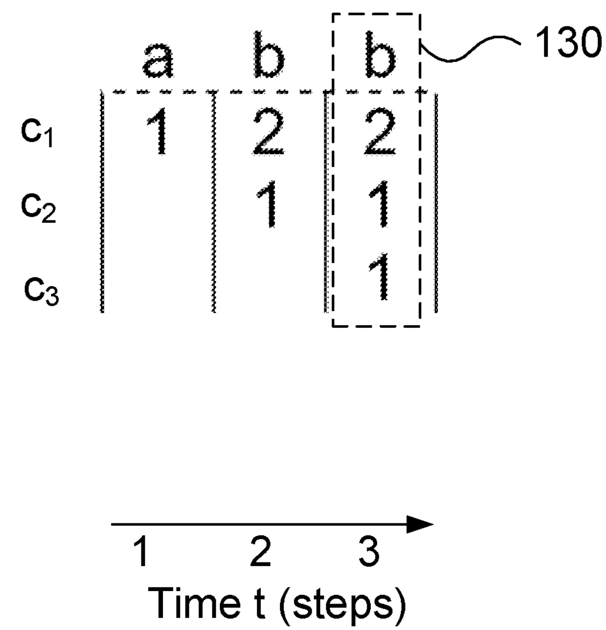

a b b
130
c1
c2
c3
1 2 2
1 1
1
1 2 3
Time t (steps)

Fig. 1b

210 215 225 230 240 245 205

Counter Set C_i	Counter $c_i^{size\ in\ K}$	A (4K)	t (2K)	B (4K)	C (4K)	u (2K)	A (4K)
C_1	c_1^{4K}	1	1	2	3	3	3
	c_1^{2K}	0	1	1	1	2	2
C_2	c_2^{4K}		0	1	2	2	3
	c_2^{2K}		1	1	1	2	2
C_3	c_3^{4K}			1	2	2	3
	c_3^{2K}			0	0	1	1
C_4	c_4^{4K}				1	1	2
	c_4^{2K}				0	1	1
C_5	c_5^{4K}					0	1
	c_5^{2K}					1	1
C_6	c_6^{4K}						1
	c_6^{2K}						0

1 2 3 4 5 6

235 220 Time step

Fig. 2

Requested Block	Time step	Stack	Stack distance
a	1	a	**
b	2	b, a	**
b	3	b, a	1
c	4	c, b, a	**
b	5	b, c, a	2
a	6	a, b, c	3
d	7	d, a, b, c	**
c	8	c, d, a, b	4
a	9	a, c, d, b	3
a	10	a, c, d, b	1

255

Stack Distance	Stack Distance Frequency
1	2
2	1
3	2
4	1
∞	4

260

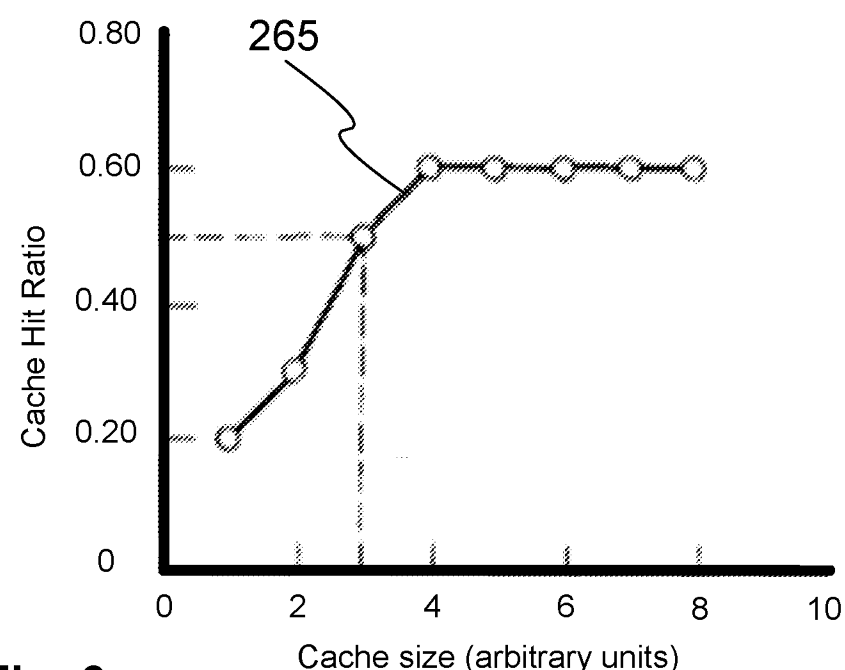
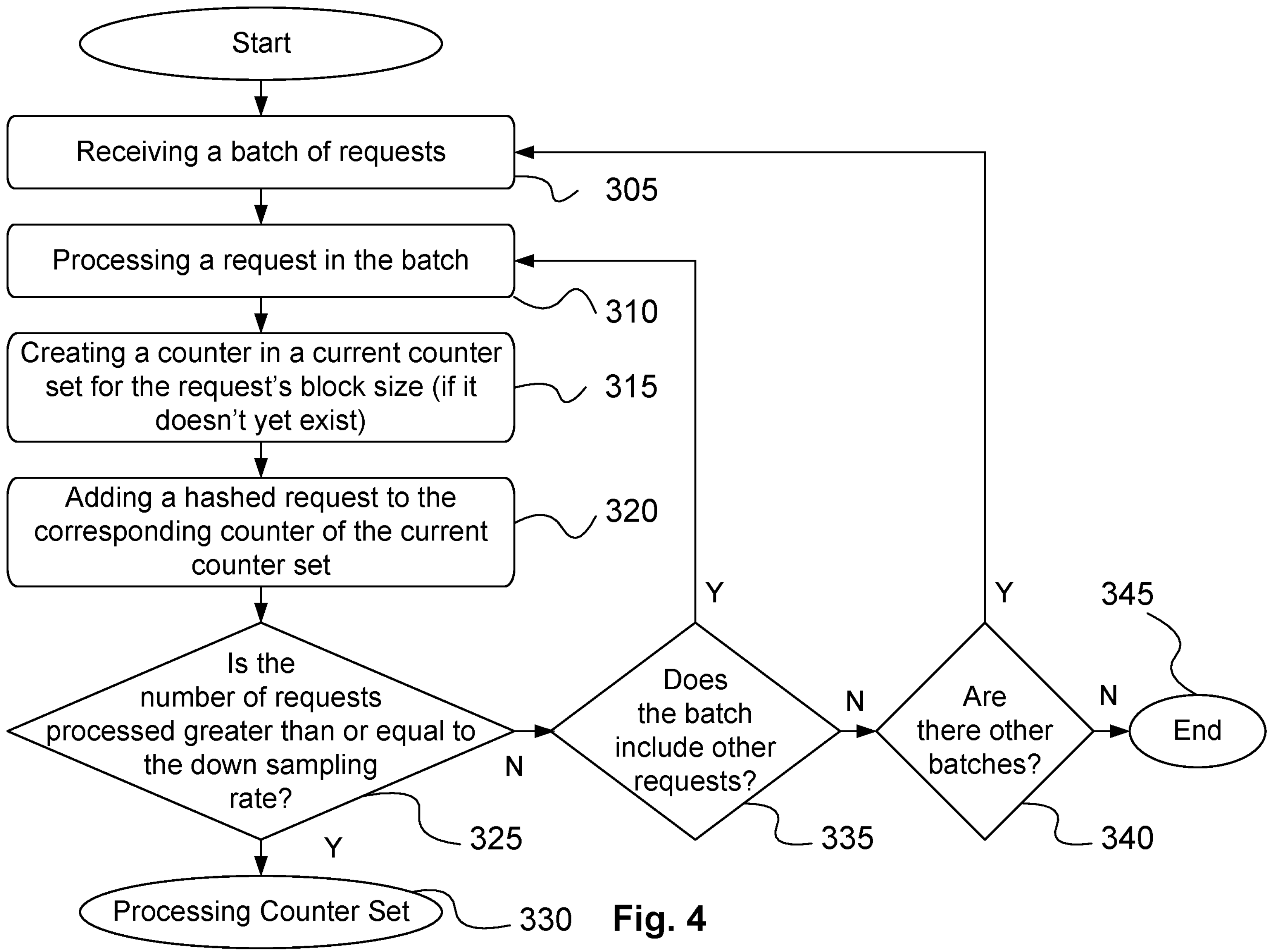


Fig. 3



Start
Receiving a batch of requests
305
Processing a request in the batch
310
Creating a counter in a current counter set for the request's block size (if it doesn't yet exist)
315
Adding a hashed request to the corresponding counter of the current counter set
320
Is the number of requests processed greater than or equal to the down sampling rate?
325
Y
N
Processing Counter Set
330
Does the batch include other requests?
335
Y
N
Are there other batches?
340
Y
N
End
345

Fig. 4

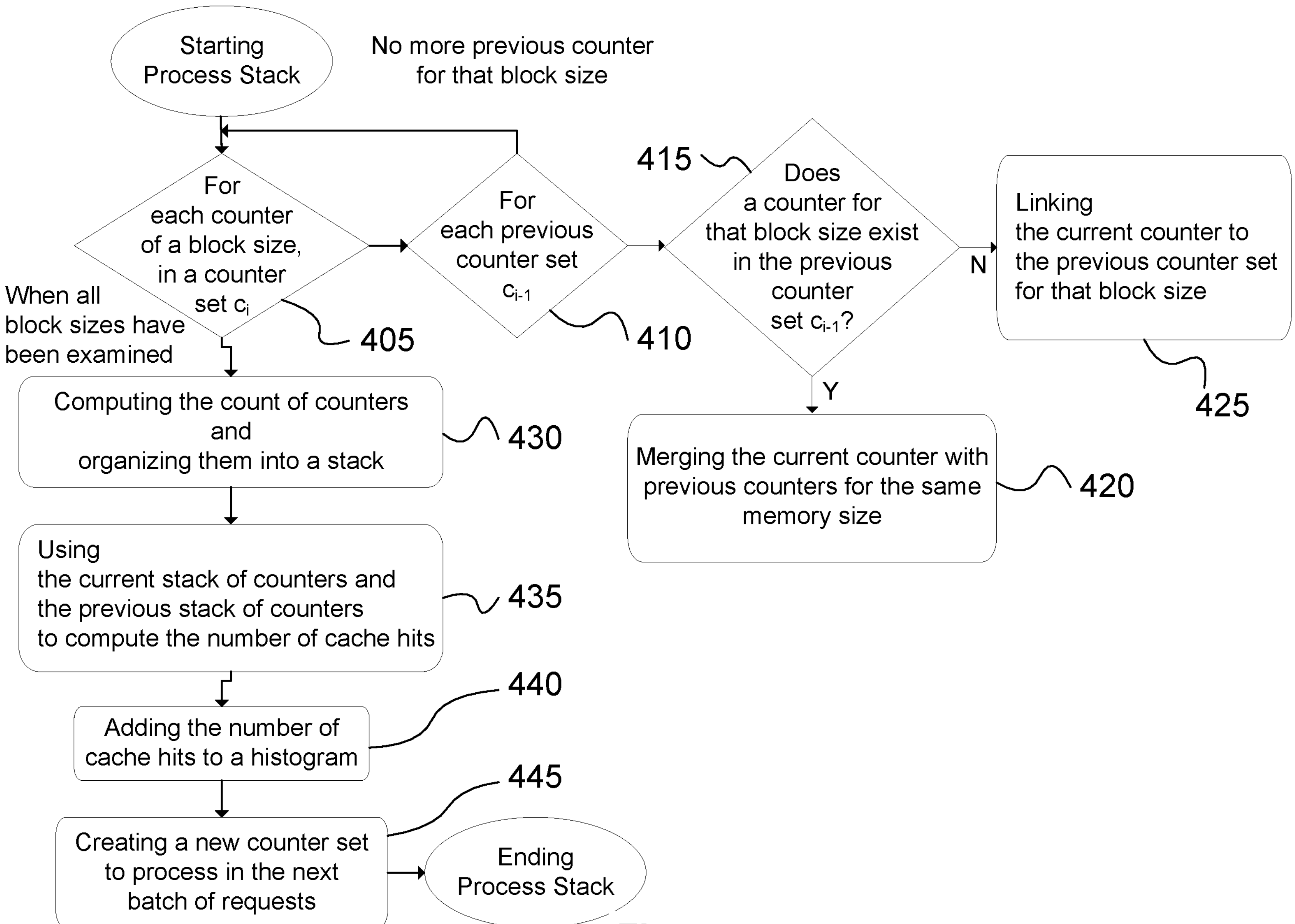
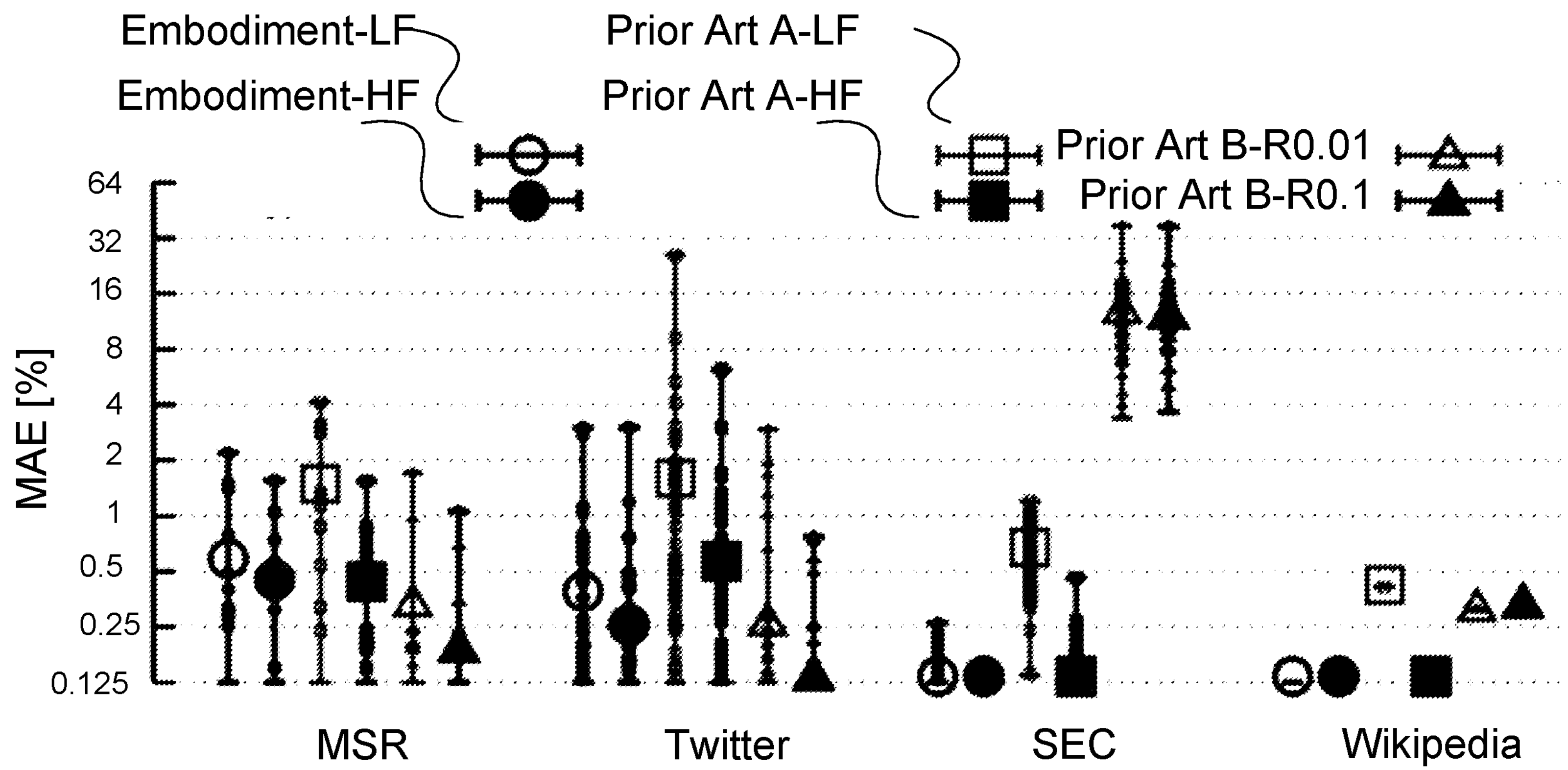

Starting
Process Stack
No more previous counter
for that block size
For
each counter
of a block size,
in a counter
set c_i
405
When all
block sizes have
been examined
For
each previous
counter set
c_{i-1}
410
415
Does
a counter for
that block size exist
in the previous
counter
set c_{i-1}?
N
Y
Linking
the current counter to
the previous counter set
for that block size
425
Merging the current counter with
previous counters for the same
memory size
420
Computing the count of counters
and
organizing them into a stack
430
Using
the current stack of counters and
the previous stack of counters
to compute the number of cache hits
435
Adding the number of
cache hits to a histogram
440
Creating a new counter set
to process in the next
batch of requests
445
Ending
Process Stack

Fig. 5

505 510

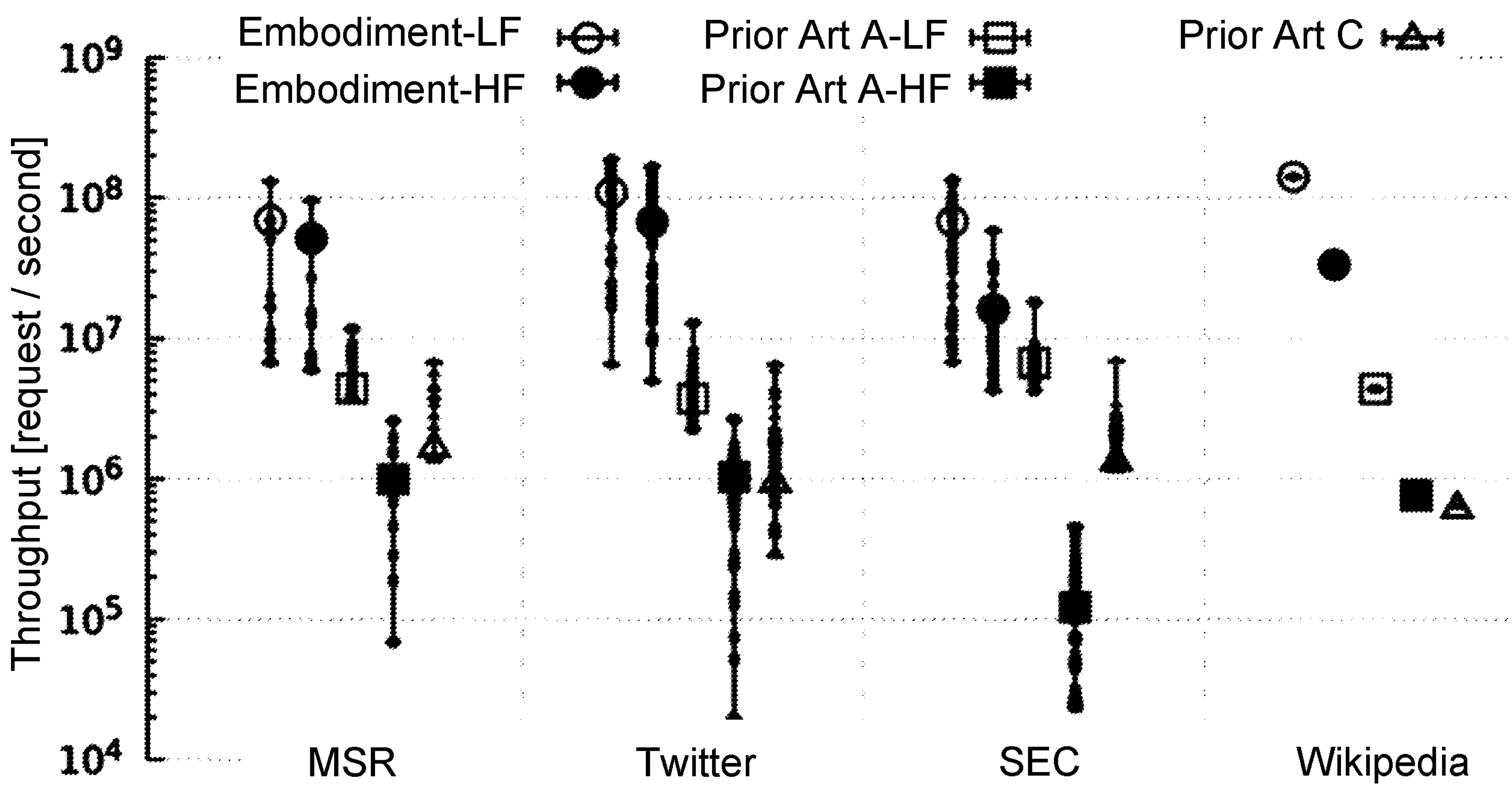
Block size	C_1	C_2	C_3
4K	$C_1^{4K} = 100$	$C_2^{4K} = 99$	$C_3^{4K} = 99$
2K	$C_1^{2K} = 1000$	$C_2^{2K} = 100$	$C_3^{2K} = 100$

Fig. 6



Embodiment-LF
Embodiment-HF
Prior Art A-LF
Prior Art A-HF
Prior Art B-R0.01
Prior Art B-R0.1
MAE [%]
64
32
16
8
4
2
1
0.5
0.25
0.125
MSR
Twitter
SEC
Wikipedia

Fig. 7



Throughput [request / second]
10^9
10^8
10^7
10^6
10^5
10^4
Embodiment-LF
Embodiment-HF
Prior Art A-LF
Prior Art A-HF
Prior Art C
MSR
Twitter
SEC
Wikipedia

Fig. 8

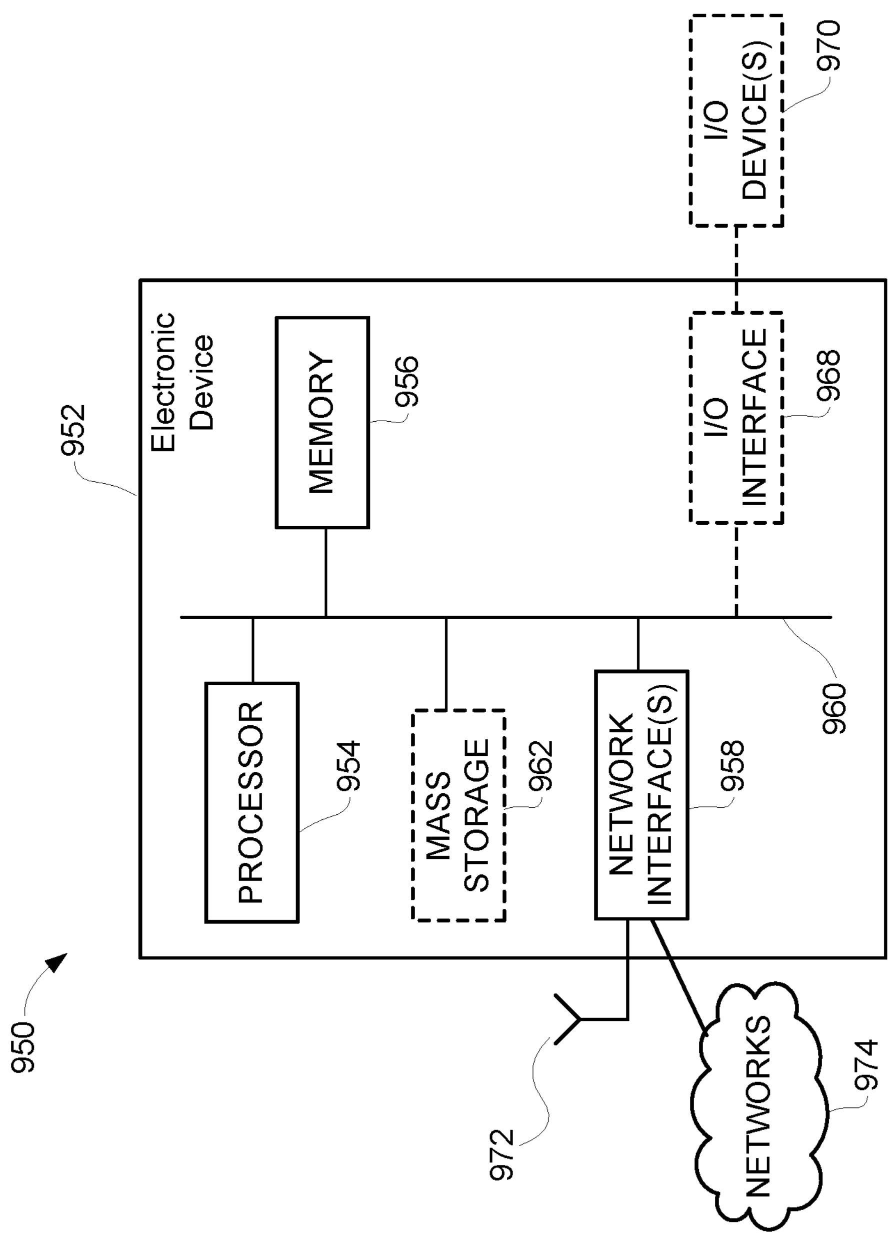

950
952
Electronic Device
MEMORY
956
PROCESSOR
954
MASS STORAGE
962
NETWORK INTERFACE(S)
958
960
I/O INTERFACE
968
I/O DEVICE(S)
970
972
NETWORKS
974

Fig. 9

Electronic Patent Application Fee Transmittal

Application Number:	
Filing Date:	
Title of Invention:	SYSTEMS AND METHODS TO GENERATE A MISS RATIO CURVE FOR A CACHE WITH VARIABLE-SIZED DATA BLOCKS
First Named Inventor/Applicant Name:	Sari SULTAN
Filer:	Randall Marusyk/Celeste Naugler
Attorney Docket Number:	92015039US01

Filed as Large Entity

Filing Fees for **Utility under 35 USC 111(a)**

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
UTILITY APPLICATION FILING	1011	1	320	320
UTILITY SEARCH FEE	1111	1	700	700
UTILITY EXAMINATION FEE	1311	1	800	800
Pages:				
Claims:				
Miscellaneous-Filing:				
LATE FILING FEE FOR OATH OR DECLARATION	1051	1	160	160
Petition:				

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				
Extension-of-Time:				
Miscellaneous:				
	Total in USD ($)			**1980**

Electronic Acknowledgement Receipt

EFS ID:	46975809
Application Number:	17982070
International Application Number:	
Confirmation Number:	6658
Title of Invention:	SYSTEMS AND METHODS TO GENERATE A MISS RATIO CURVE FOR A CACHE WITH VARIABLE-SIZED DATA BLOCKS
First Named Inventor/Applicant Name:	Sari SULTAN
Customer Number:	139050
Filer:	Randall Marusyk/Celeste Naugler
Filer Authorized By:	Randall Marusyk
Attorney Docket Number:	92015039US01
Receipt Date:	07-NOV-2022
Filing Date:	
Time Stamp:	16:28:46
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	yes
Payment Type	CARD
Payment was successfully received in RAM	$1980
RAM confirmation Number	E2022A7G29282350
Deposit Account	505187
Authorized User	Celeste Naugler

The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:

37 CFR 1.16 (National application filing, search, and examination fees)

37 CFR 1.17 (Patent application and reexamination processing fees)

37 CFR 1.19 (Document supply fees)

37 CFR 1.20 (Post Issuance fees)

37 CFR 1.21 (Miscellaneous fees and charges)

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Application Data Sheet	92015039US01-Filed-ADS.pdf	1256430 85ed3fec260aed8201a5695f04a1fc0ddd4d8ee9	no	11

Warnings:

Information:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
2		92015039US01-Filed-App.pdf	289359 2d1af7aeab50d4eb18c2f795f281d09925f3449d	yes	26

Multipart Description/PDF files in .zip description

Document Description	Start	End
Specification	1	21
Claims	22	25
Specification	26	26

Warnings:

Information:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
3	Drawings-only black and white line drawings	92015039US01-Filed-Figs.pdf	764695 e0f1874a6bdb86f56504811b3ad2c07ae8b88f9e	no	9

Warnings:

Information:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
4	Fee Worksheet (SB06)	fee-info.pdf	46595 ecaefc601aef47eaa1f9b9072c9b337d99ede3f2	no	2

Warnings:

Information:

Total Files Size (in bytes): 2357079

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.
National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.
New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.