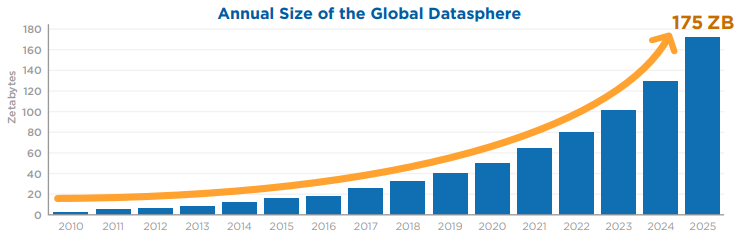
# 基于数据热度的自适应压缩存储优化研究

## 选题的来源、研究的目的意义

#### 背景

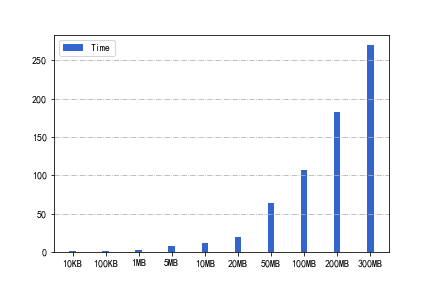
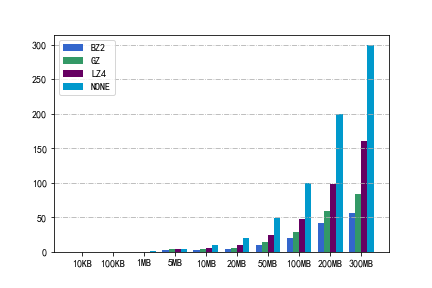
**背景**（包括在我国应用的前景）、应用价值、创新点以及国内外研究现状及水平：【课题来源，项目名称；与课题有关的国内外研究状况；】

随着8K、5G、IoT、大数据、AI等系列技术的发展，数据量迎来了爆发式增长；现代应用中的数据爆炸现象给存储系统带来了巨大的压力。此前国际数据公司（IDC）估计，全球生成的数据量将每2年翻一番。如图一所示，到2025年，全球数据将增长到175ZB【1】。**目前存储领域更偏向于在大容量、低成本与高性能相结合的存储系统的研究。**这就表明在当前大数据的背景下，不仅要求存储系统要具有高效的系统性能，还要求在较低的成本下具有持续存储的能力。其中为了能让存储系统具有强大的持续存储能力同时减少IO传输成本，利用数据缩减技术来减少数据所占用的存储空间显得尤为重要。数据缩减技术主要包括纠删码、重复数据删除、数据压缩等技术。其中数据压缩技术作为 数据缩减技术已经存在了很长时间，并且现在仍然是较为主流的数据缩减技术。

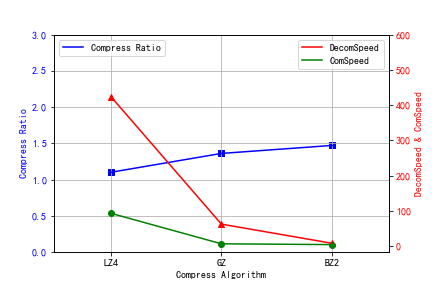
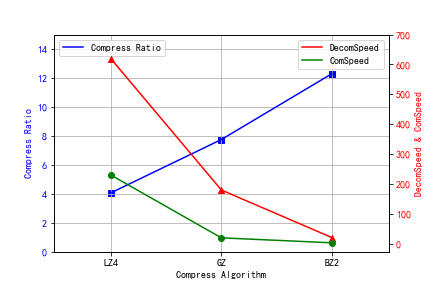


**图一: IDC预测数据量**

从压缩能力上可以将数据压缩技术分为两类：重量级压缩算法和轻量级压缩算法【20/21】；两种压缩算法在对数据的压缩率和压缩速度的表现上相差较大。如图四所示，不同级别的压缩算法在压缩率和压缩速度上的各有所长。重量级压缩算法具有良好的通用性和较高的压缩率，其缺点是解压时间较高且开销较大，适合像存档、冷备份这种对存储占用空间有较高要求的场景。在这样的背景下，为了减少压缩/解压缩所带来的开销，轻量级压缩方法在存储系统中也得到了普遍的应用。虽然轻量级压缩算法对数据的压缩率较低，但是压缩速度较快，在减少 I/O 量和数据传输量以及减少开销上有较大优势，可以有效缓解重量级压缩带来的开销问题。适合对访问速度有要求的数据场景。

****

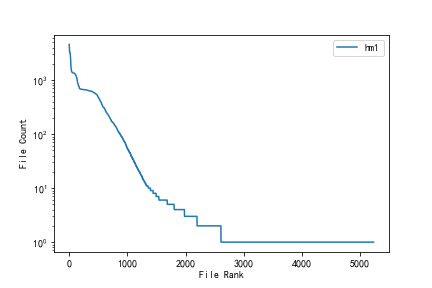
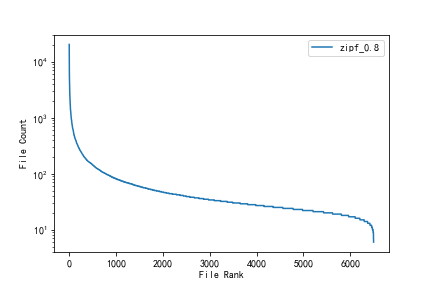
**图二、数据压缩率 图三、响应时间**

****

**图四、不同文件使用不同压缩算法的压缩性能**

不同的压缩算法有着不同的特点，通常为不同特性的数据选择最合适的压缩算法是很困难的。实际上没有任何压缩算法能够为所有类型的数据、文件格式或应用程序要求提供最佳性能。并且即使在应用程序中，应用程序的各个部分也可能有不同的压缩需求。数据压缩虽然能够有效减少存储空间，但是使用压缩算法的同时也要考虑对系统性能的影响。在工作负载不断变化的场景下，就需要根据不同的情况来对数据的压缩与否和压缩效果做决策；因此，需要从多方面考虑：数据需不需要压缩、若需要压缩，又要选择那种压缩算法既不影响总体的系统性能，又能有效减少存储空间。**问题引出。【加个对比图的解释?】**

从访问程度上来说，现实世界的数据通常有着冷热之分。据信息生命周期ILM理论，数据在存储系统中的访问次数随着时间的延长逐渐降低【】。在实际的应用场景中，尽管数据体量很大，但是大多数数据是并不活跃的数据。根据zipf定律，存储系统中的数据通常有冷热之分，互联网中大约80%的访问操作会落在20%的数据中。通常将这20%经常被访问的数据称为热数据，另外80%的一段时间没有被访问或者一直没有被访问的数据被称为冷数据。【Zhang L, Deng Y, Zhu W, et al. Skewly replicating hot data to construct a power-efficient storage cluster[J]. Journal of Network and Computer Applications, 2015, 50: 168-179.】**例如：xx系统在那段时间的访问情况如图五所示**。【Wikipedia workload analysis for decentralized hosting】从数据的访问情况来看，分别对冷热数据进行不同的压缩处理，从一定程度上能够提升系统的访问性能和存储性能。【在实际业务场景中，数据根据时间分为热，温，冷三类数据。热点数据通常查询或者更新频繁，对访问的响应时间和实时性要求较高。冷数据访问频率低，只是偶尔访问甚至不会被访问。温数据则介于两者之间。】



**图五、Zipf\_0.8 图六、MSR-Cambridge hm\_1**

因此根据数据温度和负载情况为数据分配压缩算法，并且根据不断变化的访问模式和资源负载情况实时调整数据的压缩算法来进一步优化存储系统是很重要的。根据数据热度构建弹性压缩调整算法，需要考虑不断变化的数据热度和负载情况。所以我们既不能不区分应用场景和数据特点一味的使用压缩算法来降低存储占用空间而忽略掉了压缩算法为系统访问性能带来的访问延迟的压力；也不能不区分数据访问情况不使用任何压缩算法来维持系统性能，但是却给持续存储带来压力。因此我们需要从多方面考虑，根据数据可压缩性、数据热度与资源负载情况来实现对不同场景的自适应压缩分配和实时调整数据的压缩算法来维持系统性能和可持续存储能力的平衡。

根据数据在生命周期中热度或价值的变化，针对不同阶段的数据采用恰当的压缩算法和不同的调整策略进行数据的管理，来提升系统的可持续性存储和平衡是非常重要的。

#### 国内外现状

从上述可以得知对于不同类型和价值的数据使用不同的压缩算法对系统的优化和影响相差甚大。因此自适应压缩选择算法很有必要。自适应压缩选择是指根据实际的应用场景在不同方面针对数据特征选择相应的压缩算法对数据进行压缩，从而实现存储性能和系统性能的提升。为了缓解爆炸增长的数据，使用自适应压缩选择算法来提升系统持续存储能力和系统访问能力变得尤为关键。目前自适应压缩选择算法主要是针对系统的存储性能和访问性能进行设计实现。关于自适应压缩选择算法的设计，国内外学者分别从存储性能和系统性能方面提出了相对应的压缩选择算法。下面分别针对存储性能和系统访问性能方面进行现状梳理。

###### 面向存储性能的自适应压缩

面向系统存储性能的自适应压缩选择算法涉及的目标主要是为了将数据占用的存储空间减少到最小。由于压缩算法的压缩率主要受到数据分布特征、数据类型等方面的影响，因此对于不同的数据，能获得最佳压缩率的压缩算法也可能有所不同。因此能够根据数据特性自适应选择最佳压缩算法成了进一步减少数据占用空间的一种方式。

部分学者利用数据分布特征并结合启发式算法来预测数据块的最佳压缩算法来减少数据存储占用空间。

Abadi等人【3】针对数据库(CStore)提出一种基于决策树模型的选择方法，根据列数据的不同特征利用决策树为不同的列数据选择压缩效果最好的压缩算法。但是这种方式并没有充分考虑数据分布的局部性特征。王振玺等人【4】提出一种针对分区的压缩算法选择方式，充分考虑数据局部特性，并利用分区之间的相似性来对每个分区选择最合适的压缩算法。但是相邻的分区相似性计算较为复杂。基于上述两种问题，王海艳等人【2】提出一种在数据块级别下的贝叶斯压缩策略选择方式。通过结合当前数据块的多种分布特点，相邻数据块相似性的贝叶斯压缩算法选择策略。对不同的数据块选择压缩率最佳的压缩算法，优化了HBase的存储成本。（这种方式虽然可以对数据块充分压缩，但是并没有考虑数据是否具有可压缩的价值，不能识别数据可压缩性。没有避免对不可压缩数据压缩带来的消耗）Fuzong W【9】等人从数据可压缩性的角度出发，根据设定的当前数据的可压缩性的阈值，分别为不可压缩数据、低可压缩性数据以及高可压缩性数据选择相应的压缩算法。该策略会按照数据的可压缩性来选择压缩算法，来避免不必要的压缩消耗。Y Jin等人【6】提出基于LSTM的数据块自适应压缩算法选择策略来对列数据库进行压缩存储优化。综合各种压缩算法的结合得到每类数据的最佳压缩算法。Abel D等人【7】利用强化学习的决策能力，根据数据分布特征为不同数据快选择最佳的压缩算法。还是需要更改&添加新的

以上追求极致压缩率的自适应压缩选择方式具有成熟的理论和效果，但是这些方法未能识别和利用数据的可压缩性和访问模式的多样性，只适合对于访问极不频繁数据的场景。对于访问模式不断变化的情况下，这些方法很容易影响系统访问性能。因此为了缓解极致压缩带来的系统性能的影响，出现了兼顾存储性能和访问性能的自适应压缩算法选择策略。

###### 兼顾存储性能与系统性能的自适应压缩

面对不同的场景，影响系统性能的因素也有所不同。不断变化的访问模式、资源利用率以及压缩算法的压/解压缩速度都会对系统性能造成不等的影响。为了在使用压缩算法减少存储的同时不会影响系统的访问性能，

一些学者分别考虑了不同的影响系统性能的因素，提出了一些在减少存储的同时能够进一步优化系统性能的策略。

Devarajan H等人【8】首先通过针对各种数据类型，文件类型的大量分析得出每种数据类型和文件类型所适合的最佳压缩算法。然后通过对不同访问模式/负载情况的文件分为不同的层次，并为不同层次的文件分配在压缩算法指标上的权重来计算每种类型的数据在不同压缩算法的分数。最后通过决策树模型和反馈机制实时选择最佳算法并通过反馈机制定期调整模型来改进模型精准度。这种方式可以在已知文件访问模式的情况下结合相应的压缩算法来缓解存储空间压力和对系统性能的影响。但是这种方式需要提前获取每种数据类型文件的指定访问模式，在不断变化的访问模式下的应用效果并不突出。

对于不断变化的资源利用率，数据压缩的效果也会有所变化。Mao B等人【10】针对数据的可压缩性和不同系统资源利用率对数据压缩的影响，提出了一种根据实时资源利用率自适应分配压缩算法的策略。对于可压缩数据块，该方案利用工作负载的压缩多样性，并在系统利用率较低的时期采用较高的压缩率算法，在系统利用率较高的时期采用较低的压缩率算法。对于不可压缩（或极低可压缩）的数据块，它将直接把它们写入闪存存储，而不进行任何压缩。但是这种方式没有考虑从压缩算法的压缩和解压缩时间，在某些场景中的效果并不太好，对于访问频繁的场景下很容易造成较高的访问延迟。压缩算法的效率主要受到cpu和io的影响。考虑到数据的压缩效率，Zou H等人【11】在数据传输场景下，针对cpu和缓存对压缩/解压缩时间的影响。提出一种自适应数据压缩算法来降低数据传输成本。该方案根据实时可用的系统资源来计算最佳传输的压缩算法。Lu T【15】等人提出以一种基于资源感知的自适应压缩算法选择模型。根据物联网传输数据的不同需求，在不同条件下的资源利用率的情况下，最大限度地减少从边缘到云的数据传输时间或最大限度地提高压缩比？？？？。王海艳等人【2】针对HBase没有区分冷热数据进行压缩的情况，提出一种根据冷热数据来实施不同的压缩算法。该方案使用决策树提前识别数据冷热，然后对冷数据使用重量级压缩算法，热数据使用轻量级压缩算法。有效的缓解了系统性能的影响。但是该方案并不能实时感知数据热度的变化并且没有办法针对不断变化的访问模式实时调整数据相对应的压缩算法。郭静【12】将自适应压缩选择问题构建成一个多目标优化问题来对海量归档数据进行管理优化。该方法根据存档数据的成本，对存储减少量，对系统性能的影响以及可靠性的影响构建成一个多目标优化问题，以此来管理归档数据。Huang X【18】等人针对物联网边缘设备传输数据到服务器的耗时问题，提出了一种具有任务完成时延保证的推理任务压缩比选择的传输方案。通过在感知剩余时延预算的情况下动态选择最佳压缩比，在通信资源约束下，可以及时完成更多的任务，并得到正确的推理结果。Huang S【13】等人针对Spark Shuffle过程中的网络传输时间、io读写时间以及cpu利用率和处理数据的大小密切相关的问题，提出了一种Spark Shuffle过程的成本优化模型，自适应的根据压缩算法特性和实时变化的cpu利用率、内存使用率等环境因素，为任务选择最佳的压缩配置，来提升spark在shuffle过程中的执行性能还是需要更改&添加新的

## 二、论文研究的主要内容，方案和拟采用的研究方法、手段。

已进行的科研工作基础和已具备的科学研究条件（包括文献资料及主要实验仪器设备准备情况等），对其它单位的协作要求。论文总工作量（估计），论文初稿的进度以及预期结果：【本课题研究的主要内容。技术方案（技术路线，技术措施）；存在的主要问题预期达到的目标。技术关键；】--可以画出初步构建的中间件的图。

#### 2.1研究目标

针对问题，简要说明本论文研究的学术目标。

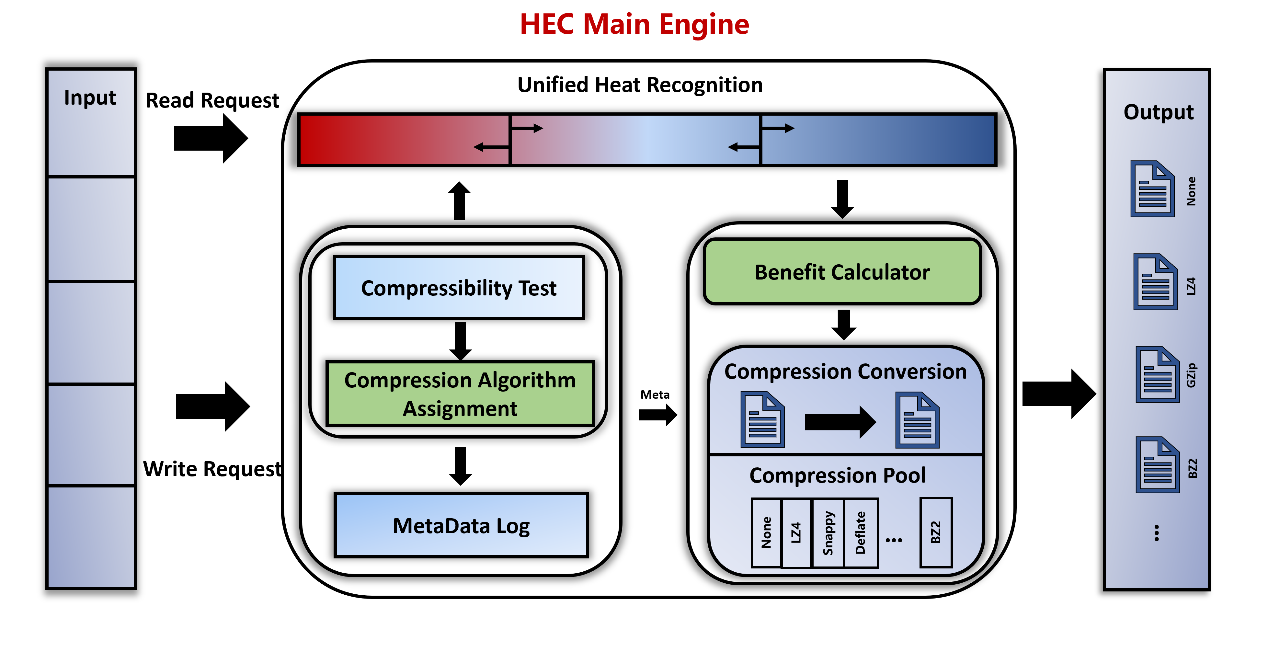
通过根据不同数据温度与资源利用的情况下，为每种数据选择合适的压缩算法。并且实时对数据的压缩算法进行及时调整来实现大数据存储的自动化管理。

**说明要研究的大览：**根据数据访问频度什么什么...构建了一种冷热分离的方式。此外在冷热分离的基础上根据文件存储系统的读写流程以及存储节点的资源利用率情况提出了一种兼顾压缩算法特点、数据温度、资源利用率的弹性压缩算法选择与调整算法。具体研究如下。

**\*\*大数据时代的到来给企业数据的存储和管理带来了新的挑战。目前存储领域更偏向于在大容量、低成本与高性能相结合的存储系统的研究。为了提升存储系统的可持续存储能力，使用数据压缩来进行数据缩减成为了一种主流的方式。然而由于不同的数据压缩算法在压缩效果和压缩速率上有着很大的差别，对所有数据使用相同的压缩算法并没有识别和利用数据的可压缩性和访问模式的显著多样性，错过了提高系统性能、空间效率或两者兼而有之的机会。因此如何为存储系统设计一种面向不同数据和场景来自适应选择压缩算法来进一步提升可持续存储能力和系统性能变得很重要。**

**根据前文梳理的研究现状，发现目前现有的自适应压缩选择机制一方面没有全面考虑到不断变化的数据访问模式以及资源利用率对压缩效率的影响；另一方面现有自适应压缩机制并没有考虑如何实时动态调整某些数据价值发生变化的数据所使用的压缩算法来进一步提升存储系统的可持续存储能力以及访问性能。因此本课题主要针对不断变化的数据热度和系统环境研究一种自适应压缩与调整的存储优化架构。主要分为一下几部分。\*\***

**HEC是一个结合自适应压缩分配与在线调整的框架。HEC主要由三个模块构建而成：1、数据热度实时监控模块，它主要是对数据进行全局热端监控，并根据工作负载的变化动态调整热区域范围的大小；2、压缩算法分配模块，利用它来识别数据的可压缩性，将可压缩数据的压缩信息写入MetaLog，并结合数据热度和压缩信息权重来计算数据在不同热度等级下可以使用的最佳压缩算法；3、在线调整模块，它首先构建成本模型，在不同热度等级下根据成本模型来判断是否需要进行更换压缩算法的操作。【还是那个根据温度的变化更换压缩？还是定期逐层往下更换压缩？？】**



**图七、整体框架**

#### 2.2主要研究内容

分要点、有层次地说明要实现研究目标所需要完成的研究内容。一般有3-4项研究任务。

(1)内容的描述。解释该内容【**是需要说明一下解决方案的**】

**提出并实现一种基于数据热度的压缩算法分配机制**

**提出并实现一种面向利益最大化 / 成本最小化的压缩算法在线调整策略**

**提出一种自适应压缩与调整的存储优化总体架构。【架构图】**

**（1）构建一种冷热数据识别机制【一种动态调节的数据热度/热数据识别机制】**

为了充分利用不同压缩算法的优势特性，需要为不同热度的数据分配不同的压缩算法；这里首先以zipf定律和信息生命周期(Information Lifecycle Management ILM)理论为指导，利用数据的访问模式，通过对数据的访问频度、时间访问特性、数据大小等方面的综合考虑。结合数据结构与实时热度衰减调整策略建立实时冷温热数据识别机制来将数据划分为冷温热三种数据【不同温度的数据】。并根据数据访问量动态调整冷热队列的大小。**基于数据结构的冷温热数据划分方式能够实时的利用数据热度对数据进行划分，提高热数据访问命中率，并为后面对数据分配压缩方式奠定了基础，为实时压缩方法调整提供了可靠的待更新数据集合**。【并根据负载的波动来动态调整温热队列的大小】

**===================================================================**

为了识别不必要的压缩，以减少不必要的计算开销和访问延迟，重要的是要有选择的压缩那些不会被频繁访问的数据，将数据进行压缩以减少存储压力；同时避免压缩那些频繁访问的数据，将数据的存储格式保存成原有的数据格式来避免访问延迟和计算开销的过分增长；因此，为了更细化的对不同访问频率的数据判断是否进行压缩，以及使用哪些数据压缩算法。HEC根据整体数据的访问情况，将数据划分为两个区域，热区域和冷区域，并对数据的热度进行实时监控。**下面说明方案：【Word里面用写详细的方案吗？？还是说简单写写就行？？】**

**说明：**

如上述所言，HEC会在选择部分数据进行压缩来缓解存储压力。选择压缩的数据越多可以节省的存储空间也就越多，但是同时也会在I/O进行耗时的解压操作。因此为了适应不断变化的工作负载，避免热区域范围过大或过小，HEC使用一个简单的访问成本模型，在数据热度识别的基础上定期根据总体访问的成本来动态的调整热区域的范围大小。**下面说明方案：**

**说明：**

***再加个扩大/缩小范围之后的访问成本吧，这样可以对比一下***

**【是不是需要和扩大或者减少之后的访问成本对比一下？】如果当前周期的访问成本大于上个周期的访问成本，则说明当前周期有较多访问命中在候选队列中，这是可以适当扩展热队列，将部分候选队列中的数据移到热队列中来适当减少访问带来的开销。否则，说明当前周期有较多的访问命中在热队列中，那么可以适当缩减热队列，将部分热队列中的数据移动到候选队列，等待压缩来适当减少存储空间的占用。【感觉还是要和扩大/缩减之后的比一下才能可以】**

**基于数据结构和动态调整的热数据识别机制能够实时的利用数据热度对数据进行划分，提高热数据访问命中率，并为后面对数据分配和调整压缩方式奠定了基础，为在线调整压缩方法提供了可靠的待更新数据集合**。

**===================================================================**

**（2）提出并实现一种压缩算法分配机制研究【数据温度/访问模式/系统资源利用率】**

针对存储系统不能有效识别数据可压缩性以及不能自适应的根据数据访问模式和系统环境为数据选择最合适的存储方式。提出一种结合数据热度、访问模式和系统环境方面的自适应压缩分配策略。有效地利用不同压缩算法的性能，从而充分评估数据在每种压缩算法的压缩率、压缩时间以及解压缩时间。进一步地，通过识别数据访问模式以及数据热度自适应分配每个数据在压缩率、压缩时间、解压缩时间所占比重来建立压缩算法总体性能评估模型。最后利用模型计算每个数据在不同压缩算法上的表现分数，选取最佳的压缩算法。

自适应压缩分类策略为不同场景下的数据分配最合适的压缩算法，可以有效地将低温度数据进行不同程度的压缩来减少数据占用的存储空间，同时为高温度的数据选择拥有最佳访问性能的存储方式。可以有效地提升存储系统的可持续存储能力以及系统性能。

===================================================================

还是说把前两条合并成一个

不同资源利用率下，适合的压缩方法不同；不同热度下，适合的压缩方法不同；

不同的压缩算法有着不同的特点和应用场景。目前并不存在能有一个压缩算法能够适应所有的场景。因此为了有效缓解存储压力并提升系统性能使两者达到平衡，这里需要结合存储系统的资源利用率、数据热度以及不同压缩算法的特性来为数据分配相对应的压缩算法。【先说明一下资源利用率[cpu io mem]的影响】【然后说明为不同热度的数据分配压缩算法的好处能达到的目标】

数据中总是存在着不可压缩的数据。通过数据取样进行测试的方式预估数据的可压缩性，在确定过数据的可压缩性之后，综合考虑数据的访问模式，为不同的数据自适应分配最合适的压缩算法；

数据的访问模式和热度不同，对于压缩算法的偏好自然也会不同【例如：对于热数据来说更偏向于访问性能更好的算法；而对于冷数据来说，则更偏向于压缩性能更好的算法】。压缩算法的性能评估主要由压缩率、压缩时间以及解压缩时间三种因素来进行评估。数据对于压缩算法性能的偏好，主要体现在这三种因素的权重上。分配策略主要结合数据的访问模式和热度为数据自适应的分配这三者的权重，通过三者的结合选出最合适的压缩算法。

影响压缩性能的因素主要涉及cpu、io以及内存的占用率。不同的系统资源利用率下，同一个目标下的最合适的压缩算法可能会有所不同，因此需要结合cpu、io以及内存的占用率来预估出在当前资源利用率下每种压缩算法在压缩速度和解压缩速度上的变化，来进一步优化自适应分配压缩算法的策略/来更精准的唯每个数据分配当前情况下最合适的压缩算法。

**自适应温度与负载情况的压缩算法分配方式**

**（3）提出并实现一种数据压缩算法实时调整策略研究【更换时机以及更换对象的选取】**

/\*\*在实际情况中数据的热度是在不断变化的。根据信息生命周期管理(ILM)，热数据的访问频率会随着时间逐渐降低，冷数据也存在升温变为热数据的情况。因此随着数据温度和负载情况的变化及时的为数据动态调整压缩算法可以进一步提升自适应压缩存储对系统的贴合性？因此提出一种面向利益最大化的数据压缩算法实时调整策略，根据数据温度与负载情况选择需要更换压缩算法的数据和进行更换压缩的时机。\*\*/

在(2)中的自适应压缩存储即为不同热度数据提供不同的压缩算法，具有不同方面的性能提升。热数据使用轻量级压缩算法或者不使用压缩算法，能够提供较好的访问性能，但是存储占用较高。冷数据则相反。

在许多研究中表明，系统用户生成的数据通常在其生命周期的早期获得相对较高的访问频率，而访问的总体趋势是下降的。但是并不能保证内容变冷之后不会再次变热。因此当此类数据在存储系统中最初可以使用轻量级压缩，然后数据降温之后更换为重量级压缩来优化存储成本。当数据升温之后将压缩算法转换为轻量级压缩或者不压缩来提升访问性能。然而，更换压缩的成本是不可忽略的，更换压缩之后数据的访问量可能会意外增加，这就表明草率的进行更换压缩算法将会产生更多的成本，而不会节省成本。为了做出最佳的更换压缩算法的决策，这里提出一种面向利益最大化的数据压缩算法实时【动态，在线】调整策略，【构造代价模型并基于代价模型提出一种数据压缩算法在线调整策略】。根据数据温度与负载情况【结合数据的未来访问概率】选择需要更换压缩算法的数据和进行更换压缩的时机。基于不同数据使用不同压缩算法在访问和存储两方面的利益对比来识别是否需要进行更换压缩算法。【分别对数据升温和数据降温采用不同的调整策略。要不要具体说下升温和降温的策略。。。】【参考了Keep\_Hot\_or\_Go\_Cold\_A\_Randomized\_Online\_Migration\_Algorithm\_for\_Cost\_Optimization\_in\_STaaS\_Clouds】--具体的拟用方法。

===============================================================================

不能热度一变化就进行算法更换。需要考虑更换代价与利益、当前负载情况。【更换条件1&2】减少不必要的操作。

根据信息生命周期管理(ILM)，曾经经常被访问的数据的数据热度会随着时间的延长逐渐降低/冷数据可能也会升温变为热数据。在这种不停变化的访问模式和数据热度中，需要为有些数据的压缩算法进行实时的更换，来提升系统性能。

根据上述研究内容来说，数据的热度不断变化，不同热度的数据要使用的压缩算法也有所不同。当数据热度变化的时候，需要对数据进行压缩算法更换。那么根据访问模式和系统负载情况选择数据更换压缩算法的时机以及需要更换压缩算法的数据对象的选择尤为重要。既要准确的根据变化的数据热度进行压缩算法的调整也要根据系统资源的实时负载对数据的压缩算法进行调整。为了准确的抓住更换时机，同时也为了避免不必要的压缩算法更换，在这里提出一种面向利益最大化的数据压缩算法实时调整策略。根据这一策略实现实时弹性压缩调整策略，提升系统的总体存储性能和系统性能。

短时间内来回变化的说明

**动态阈值调整**

（4）混合存储介质选择与迁移策略？

/数据对象在其生命周期内可能会根据其特征具有不同的访问模式。

一些活动对象在其生命周期中经常被访问和修改。此类对象处于热状态。其他的可以在开始时频繁访问（即，热状态），然后随着时间的推移（即，冷状态），对对象的访问速率显著下降。这种模式通常在在线社交网络（OSN）中观察到（Muralidhar等人，2014年），这些网络是本工作的参考领域。因此，将数据存储在针对特定访问频率优化的存储层中是经济高效的/

/热层具有较高的存储成本，但访问成本较低，因此更适合存储频繁访问的对象。

相比之下，冷层具有更低的存储成本和更高的访问成本，从而使其更适合存储不经常访问的对象。

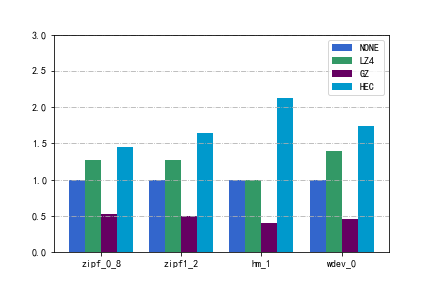
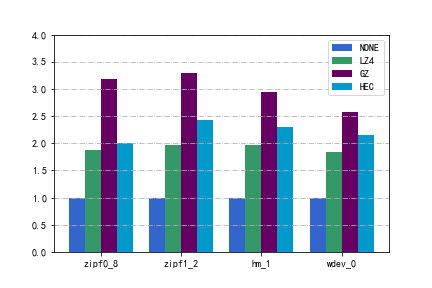
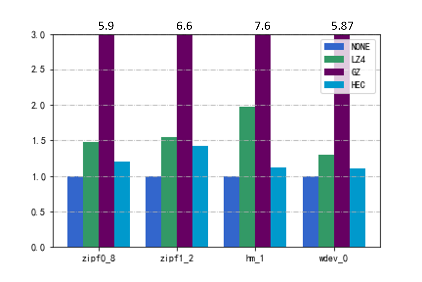
【替换成不压缩和压缩】；决定对象在热层或冷层中的最佳位置在存储服务的成本管理中起着至关重要的作用。

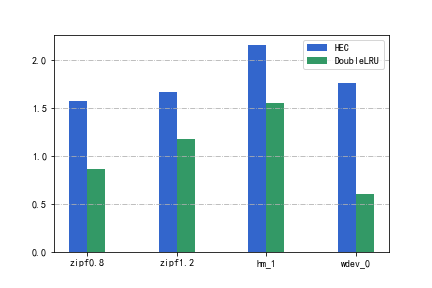
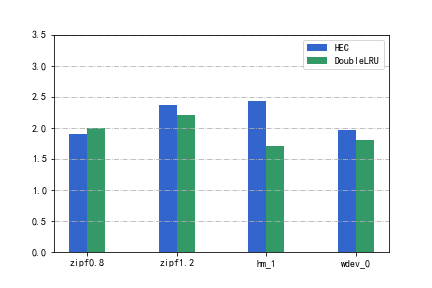
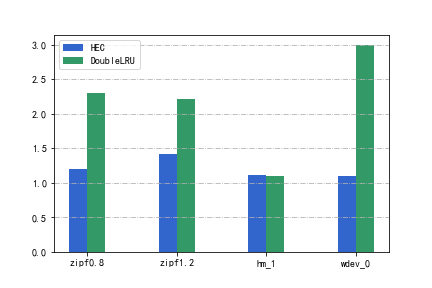
简单地在对象的整个生命周期中使用热层可能是低效的。考虑存储一个30 GB的blob（由许多对象组成）**一个举例具体说明，比如全部不压缩的访问和存储情况，全都压缩的访问和存储占用的情况**

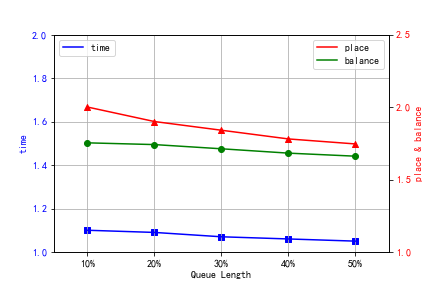
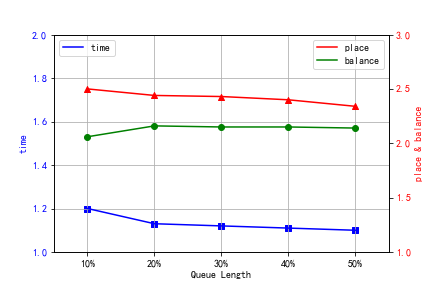
这个简单的示例表明，始终将数据仅保留在一个层中是低效的，尤其是对于时变的工作负载。

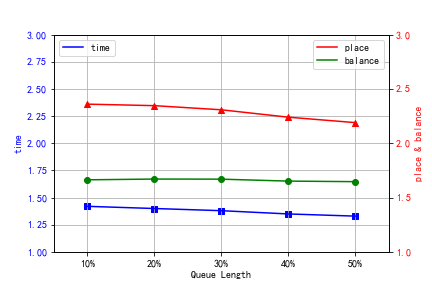
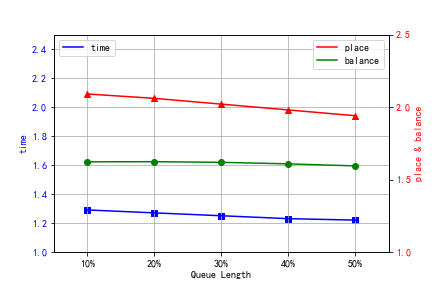
因此，考虑到时变的工作负载和存储和访问定价相反的两层存储服务，出现了两个问题：（i）在对象生命周期的每个时隙中应使用哪一层？和（ii）对象何时应从热层转移到冷层，反之亦然？【转换成不压缩与压缩】

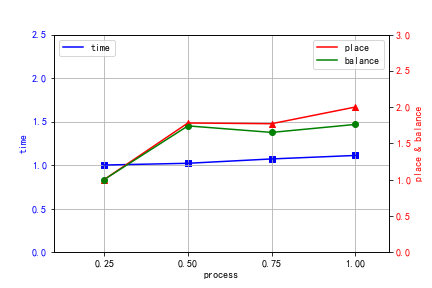
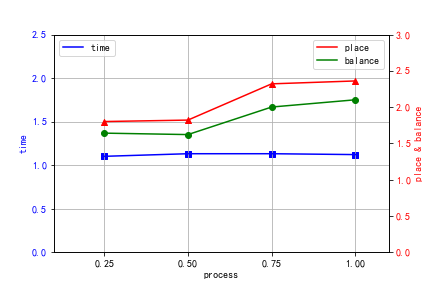
初步实验结果

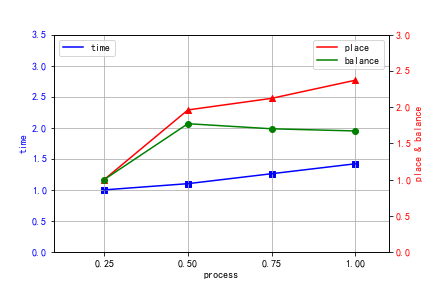
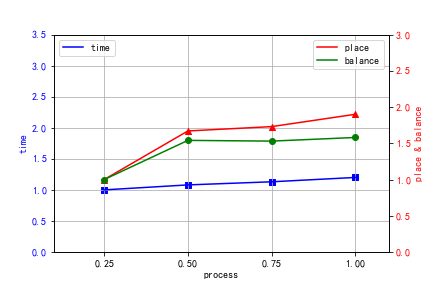












参考文献

【1】W. Shi, G. Pallis and Z. Xu, "Edge Computing [Scanning the Issue]," in Proceedings of the IEEE, vol. 107, no. 8, pp. 1474-1481, Aug. 2019, doi: 10.1109/JPROC.2019.2928287.

【2】王海艳, 伏彩航. 基于 HBase 数据分类的压缩策略选择方法[J]. 通信学报, 2016, 37(4): 12-22.

【3】Trondheim N, Stonebraker M, Abadi D J. C-store-A column-oriented DBMS[C]//The 31st VLDB Conference. Trondheim, Norway, c2005. 2005: 553-564.

【4】王振玺，乐嘉锦，王梅等.列存储数据区级压缩模式与压缩策略选择方法[J].计算机学报，2010，33(8):1524-1530.

【5】Yu X, Peng Y, Li F, et al. Two-level data compression using machine learning in time series database[C]//2020 IEEE 36th International Conference on Data Engineering (ICDE). IEEE, 2020: 1333-1344.

【6】Jin Y, Fu Y, Liu T, et al. Adaptive Compression Algorithm Selection Using LSTM Network in Column-oriented Database[C]//2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC). IEEE, 2019: 652-656.

【7】Abel D, Arumugam D, Asadi K, et al. State abstraction as compression in apprenticeship learning[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2019, 33(01): 3134-3142.

【8】Devarajan H, Kougkas A, Sun X H. An intelligent, adaptive, and flexible data compression framework[C]//2019 19th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID). IEEE, 2019: 82-91.

【9】Fuzong W, Helin G, Jian Z. Dynamic data compression algorithm selection for big data processing on local file system[C]//Proceedings of the 2018 2nd International Conference on Computer Science and Artificial Intelligence. 2018: 110-114.

【10】Mao B, Wu S, Jiang H, et al. EDC: Improving the performance and space efficiency of flash-based storage systems with elastic data compression[J]. IEEE Transactions on Parallel and Distributed Systems, 2018, 29(6): 1261-1274.

【11】Zou H, Yu Y, Tang W, et al. Improving I/O performance with adaptive data compression for big data applications[C]//2014 IEEE International Parallel & Distributed Processing Symposium Workshops. IEEE, 2014: 1228-1237.

【12】郭静. 海量数据归档系统数据压缩算法选择方法研究[J]. 收藏, 2019, 7.

【13】Huang S, Xu J, Liu R, et al. A novel compression algorithm decision method for spark shuffle process[C]//2017 IEEE International Conference on Big Data (Big Data). IEEE, 2017: 2931-2940.

【14】Chakraborty P, Tharini C. Integration of Prediction Based Hybrid Compression in Distributed Sensor Network[J]. Wireless Personal Communications, 2022, 122(1): 229-241.

【15】Lu T, Xia W, Zou X, et al. Adaptively Compressing {IoT} Data on the Resource-constrained Edge[C]//3rd USENIX Workshop on Hot Topics in Edge Computing (HotEdge 20). 2020.

【16】罗成,崔勇,林予松.一种基于带宽预测和自适应压缩的容器迁移方法[J/OL].计算机工程:1-9[2022-02-19].DOI:10.19678/j.issn.1000-3428.0061758.

【17】Wang R, Wang C, Zha L. PACM: A prediction-based auto-adaptive compression model for HDFS[C]//2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW). IEEE, 2016: 1617-1626.

【18】Huang X, Zhou S. Latency guaranteed edge inference via dynamic compression ratio selection[C]//2020 IEEE Wireless Communications and Networking Conference (WCNC). IEEE, 2020: 1-6.

【19】Agarwala S, Jadav D, Bathen L A. iCostale: adaptive cost optimization for storage clouds[C]//2011 IEEE 4th International Conference on Cloud Computing. IEEE, 2011: 436-443.？？？？？

【20】Sun J, Lu T. Optimization of column-oriented storage compression strategy based on HBase[C]//2018 International Conference on Big Data and Artificial Intelligence (BDAI). IEEE, 2018: 24-28.

【21】Damme P, Ungethüm A, Hildebrandt J, et al. From a comprehensive experimental survey to a cost-based selection strategy for lightweight integer compression algorithms[J]. ACM Transactions on Database Systems (TODS), 2019, 44(3): 1-46.