一、概述和问题的分析、建模

1.背景：①Video analytics queries have many implementation options which impact their resource demands and accuracy of outputs. Different implementations have different resource demands and produce outputs of varying accuracies.

视频分析请求通常是由许多部件（component）组成的流水线（pipeline）。其中每一个部件各自有多种不同的实现，其共有相同的抽象，但具体实现方式不同（类似抽象类、接口这样）。如物体追踪的探测器，可以用背景减法做，也可以用DNN做。

②While the accuracy of a query depends only on its plan, its network and CPU resource demands are determined by component placement across the hierarchy of clusters.

（We define query planning as selecting the best combination of implementations and

knob values for a query.）

③multiple queries analyzing video from the same camera often have common components.

来自同一相机源的不同请求间往往需要共同的组件完成。

一条查询的准确率只取决于其选择的实现方式和配置，而其对网络、CPU等资源的需求还取决于流水线组件在分层集群中的分布。因此，决定一个合适、可取的placement也很重要。

2.目的： Our objective is to select the “query plan” – implementations (and their knobs) – and place it across the hierarchy of clusters, and merge common components across queries to maximize the average query accuracy.

(Our objective is to build a video query optimizer that takes as input: a) pipeline of components along with the different implementation options and knobs, b) cluster hierarchy with resource capacities, and c) representative video that can be used to estimate component cost and accuracy. The optimizer then automatically, (i) determines the query plan for each video query, (ii) places its components across the hierarchy of clusters, and (iii) merges common components across queries, to maximize the average query accuracy.)

不同实现，其资源占用和准确率都不同。选择最合适的组件实现方式及其对应的配置（各自knob的值）（解决①）并适当地部署到分层集群上（解决②），以达到最大准确率和相对较少的资源耗费。同时通过单例设计模式，跨请求地共享组件，进一步减少资源耗费（解决③，**尽管这样会迫使不同请求选择相同的plan和placement**）

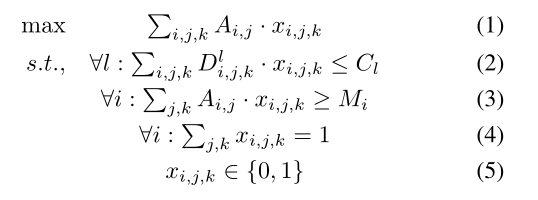
关键词：**Plan, Placement, Merge**

3.与videoStorm对比：VideoStorm [69], selects video query knobs to maximize accuracy, but ignores three important issues that we tackle. First, VideoStorm assumes that all the videos are streamed into a single cluster and hence ignores component placement in hierarchical

settings. As a result, it does not deal with multiple resources (network and compute) or dynamic network bandwidths. Second, it does not identify the opportunity to merge common components across queries. Finally, it does not consider implementation choices of vision components (only knobs).

VideoStorm只考虑视频流入单个集群，因而不会考虑集群分层间的网络带宽资源，同时也不用考虑组件的放置问题（放在哪一层）；此外，也没有考虑相同组件的合并复用问题。VideoEdge是对VideoStorm的改进。

4.Binary Integer Program 0-1规划：不考虑组件合并，可将该问题刻画为一个0-1规划最优问题：



求解该BIP需要指数时间复杂度，故不是理想的在线解决方案。

5. multiple-choice multi-dimensional knapsack problem (MMK) 多重多维背包问题：

理解：常规的多维背包问题：

①有一个背包，其在n个维度上的容量为 [公式] 。这n个维度可以包括背包载重、背包体积等等属性。

②有m件物品，每件物品i在n个维度上的“重量”为 [公式] ，每件物品i的价值为 [公式] 。

③需要从这m件物品中进行挑选并放入背包，使得物品的总“重量”不超过背包的容量，并让背包中物品的价值最大化。

而多重多维背包问题则是指，每件物品还有多种不同的化身，每个化身的重量会有所改变，其价值也会有所改变。

当前情景下，一个查询就是一个物品，查询的不同配置（实现、放置等）都是一个化身；价值就是查询化身的准确率；n维重量就是n种资源的总额限制。

但即便转化为背包问题，依然需要求解指数级时间，且没有考虑组件合并。

解决：1）使用贪心算法 2）将同一摄像头上的查询归类为super query来进行合并。

二、问题的具体求解

1.第一步：为所有请求决定实现、配置、位置

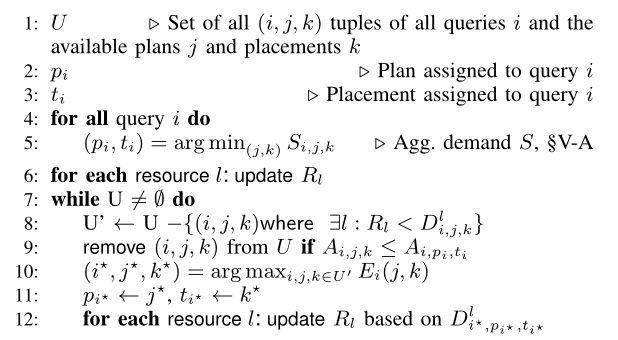
①采用主要资源需求Dominant Resource Demand

**与VideoStorm以效用方程表达请求的收益并作为选择指标不同**，VideoEdge选择通过定义资源占比Si,j,k = maxl Dli,j,k/Cl，以资源占比最大的资源需求作为主要资源需求，并作为为query选择配置的指标。

好处：1）考虑需求最大的资源，可以避免单一资源的浪费。例如，当集群间带宽不足时，多余的CPU资源会被浪费；而如果我们考虑需求最大的资源，即带宽，若优先满足当前请求对带宽的需求，则数据可以传递，CPU就不会被浪费了； 2）通过归一化得到的无量纲可以方便地扩展到多维。？？？

问题：Also, by being dimensionless, it easily extends to multiple resources, akin to DRF [31].怎么理解这个好处？

②采用贪心算法求解背包问题



这里的思路和VideoStorm差不多，但其采用的标准是资源占比而非效用大小。首先对每一请求分配最低资源占比的配置，再逐次进行遍历：每次遍历剔除资源占用超过限制的配置（保证配置可行）和对于同一请求来说没有增大效率（准确率的增量除以资源占比的增量）的配置（保证对于同一请求，最终一定得到的是唯一一个能使其达到主要资源需求的配置），直到资源耗尽或配置穷尽。时间复杂度为O（（m\*n）^2），小于指数级。

不用担心公平问题，因为一次遍历，各请求得到的资源占比增量是个小值。

2.第二步：为所有请求进行组件合并以重新分配

（When there are multiple queries processing the same camera feed with common prefix in their pipeline, we have the opportunity to eliminate running redundant components.

We refer to such queries as a peer set）

①假设前提：1）不进行部分合并，即要么不合并，要合并就合并所有组件；2）不对不常见的组件进行合并。

②在原算法的第10行，当做出配置的转变时，将此转变同时应用给i的peer queries。

③带来问题：再次增大时间复杂度。为了加速算法，通过**帕累托带**降低搜索空间U。帕累托带是由帕累托边界和δ帕累托边界两条边界线构成的带状区域，将落在该区域内的配置作为搜索空间，可大大降低时间。

理解为何要是帕累托带而不是用帕累托边界就行了：此处的帕累托最优是以主要资源考虑的，而主要资源考虑的是单个查询的某一资源的占用量及该资源容量比值。当考虑联合优化多个查询时，这些查询都采用对各自而言最大的资源，会导致该资源不够用。

3.第三步：满足预算

可以按照给出的预算量，创建一个新的虚拟资源预算并同其他资源一起应用给VideoEdge中，保证像网络带宽这样的耗费与使用成正比的额外资源消耗也在预算限度内。

问题：没找到他是怎么处理合并冲突的

问题：没看懂他和videostorm的比较。VideoStorm不是单集群的吗，哪来的放置选择？（Camera,Cluster,Cloud）（Placement Decisions Figure12）

和变色龙相比：Chameleon [35] is the recent video analytics work for continuously adjusting DNN configurations to optimize accuracy or reduce resources costs based on the

temporal and spatial correlation among the video frames. Such techniques could also be applied to our work, while Chameleon does not address query merging opportunity, which contributes to significant gains in accuracy as shown in our work.