Real-time GDA

ABSTRACT

ACM Reference format:
. 2016. Real-time GDA. In *Proceedings of ACM Conference, Washington, DC, USA, July 2017 (Conference'17),* 3 pages.
DOI: 10.475/123_4

1 CLARINET

1.1 Given claims

The paper's introduction gives the following claims which need to be investigated:

(1) Formulating an optimal solution for a multi-query, network-aware, joint query planning/placement/scheduling is computationally intractable.
✓ Cites two sources that even for a single query with the given scheduling assumptions is NP-hard.
(2) CLARINET does joint multi-query planning, task placement, and task scheduling. It picks the best WAN-aware QEP/task placement/take scheduling per query and provides "hints" to the query execution layer.
✓ "hints" are location and start times of each task.
(3) They show how to (heuristically) compute the WAN-optimal QEP for a single query which includes task placement and task scheduling. The solution relies on reserving WAN links. This is an effective heuristic for the best single-query QEP, that decouples placement and scheduling.
(4) They allow for cross-query (heuristic) optimization of n queries. They order the queries by optimal QEP expected completion time. Then they choose the ith query's QEP considering the WAN impact of the $i-1$ preceding queries.
(5) They heuristically compact the schedules tightly in time by considering the above order and groups of $k \leq n$ queries.
(6) They extend the above heuristic to accommodate (i) fair treatment of queries, (ii) minimize WAN bandwidth costs, and (iii) online query arrivals.
(7) Combats resource fragmentation and optimizes average query completion times.

1.2 Model

There are n queries and each query j has a set QEP-Set QS_j from which one QEP must be chosen. Also chosen are the task locations and task start times.

Scheduling assumptions:

(1) Network transfers do not overlap. In fact, they have a theorem that proves that any optimal schedule has an equivalent non-overlapped schedule.
(2) Obtain non-interruptible transfer schedules.

2 SINGLE DAG QUERY

2.1 Model

Let $\mathcal{L}$ be the set of sites that holds data and runs tasks. For each inter-site WAN link, let B_{ij} be the bandwidth from site $i \in \mathcal{L}$ to site $j \in \mathcal{L}$. We assume that the bandwidths are stable within the time frame of doing real-time data analytics.

Let $\mathcal{N}$ be the set of nodes and $\mathcal{D}$ be the directed set of edges respectively, for the DAG of a Query Execution Plan (QEP). We define D_i^k as the amount of input data for node k at site i, and r_j^k as the fraction of tasks for node k assigned to site j. We denote α^k as the selectivity for node k which is the proportional data shrinkage/expansion from placed tasks at each of the sites. For each node l in the DAG that has predecessors, the sum of the predecessor nodes' output data becomes the input data:

$$\sum_{j\in\mathcal{L}} D_j^l = \sum_{k:(k,l)\in\mathcal{D}} \alpha^k \sum_{i\in\mathcal{L}} D_i^k. \tag{1}$$

We assume that each node in the QEP requires a data shuffle which means that

$$D_j^l = \sum_{k:(k,l)\in\mathcal{D}} \alpha^k r_j^k \sum_{i\in\mathcal{L}} D_i^k. \tag{2}$$

We also have that the total WAN usage is:

$$\sum_{k\in\mathcal{N}} \sum_{i\in\mathcal{L}} \sum_{j\in\mathcal{L}:j\neq i} D_i^k r_j^k \tag{3}$$

which can like previously be rearranged to

$$\begin{aligned} \sum_{k\in\mathcal{N}} \sum_{i\in\mathcal{L}} \sum_{j\in\mathcal{L}:j\neq i} D_i^k r_j^k &= \sum_{k\in\mathcal{N}} \left(\sum_{i\in\mathcal{L}} D_i^k - \sum_{j\in\mathcal{L}} D_j^k r_j^k \right) &(4)\\ &= \sum_{k\in\mathcal{N}} \sum_{j\in\mathcal{L}} D_j^k (1 - r_j^k) &(5) \end{aligned}$$

2.2 Problem

Minimize WAN usage. When minimizing WAN usage for a DAG we have the following optimization problem:

$$\begin{aligned} \min_{r,D} \quad & \sum_{k\in\mathcal{N}} \sum_{j\in\mathcal{L}} D_j^k (1 - r_j^k) && (6a)\\ \text{s.t.} \quad & \sum_{j} r_j^k = 1 \quad \forall k \in \mathcal{N} && (6b)\\ & r_j^k \geq 0 \quad \forall j \in \mathcal{L}, \forall k \in \mathcal{N} && (6c)\\ & D_j^l = \sum_{k:(k,l)\in\mathcal{D}} \alpha^k r_j^k \sum_{i\in\mathcal{L}} D_i^k \quad \forall l \in \mathcal{N} && (6d) \end{aligned}$$

Even without adding deadlines, this problem has nonlinear terms in the objective function and the last equality constraint.

3 SINGLE MAPREDUCE QUERY

3.1 Model

Let $\mathcal{L}$ be the set of sites that holds data and runs tasks. For each inter-site WAN link, let B_{ij} be the bandwidth from site $i \in \mathcal{L}$ to site $j \in \mathcal{L}$. We assume that the bandwidths are stable within the time frame of doing real-time data analytics.

We perform the map tasks at the sites that contain the associated data and denote D_i as the output data from all of the map tasks at site i. The fraction of reduce tasks assigned to site j is denoted as r_j which is also the fraction of all other sites' data that must be transfered through the WAN to j. This means that the total WAN usage is for a given task distribution r is:

$$\sum_i \sum_{j \neq i} D_i r_j \tag{7}$$

which can be rearranged to:

$$\sum_i \sum_{j \neq i} D_i r_j = \sum_i D_i \sum_j r_j - \sum_j D_j r_j = \sum_i D_i - \sum_j D_j r_j \tag{8}$$

$$= \sum_j D_j (1 - r_j) \tag{9}$$

If we are given a start time s after the map steps are all completed and a data shuffle deadline t for the reduce tasks, then the completion time is bounded as such:

$$s + \max_{(i,j):j \neq i} \left\{ \frac{D_i r_j}{B_{ij}} \right\} \leq t \tag{10}$$

which is caused by the heterogeneous WAN bandwidth and has a bottlenecking link(s). We assume that $t > s$.

3.2 Problem

Minimize WAN usage. When minimizing WAN usage for a MapReduce data shuffle we have the following optimization problem:

$$\min_r \quad \sum_j D_j (1 - r_j) \tag{11a}$$

$$\text{s.t.} \quad \sum_j r_j = 1 \tag{11b}$$

$$r_j \geq 0 \quad \forall j \tag{11c}$$

Feasibility to meet deadline. We want to find a feasible r so that the data shuffle finishes at or before t:

$$\text{find } r \text{ s.t.} \quad \frac{D_i r_j}{B_{ij}} \leq t - s \quad \forall (i,j) : j \neq i \tag{12a}$$

$$\sum_j r_j = 1 \tag{12b}$$

$$r_j \geq 0 \quad \forall j \tag{12c}$$

Minimize WAN usage given a shuffle deadline. When minimizing WAN usage for a MapReduce data shuffle for a given deadline t we have the following optimization problem:

$$\min_r \quad \sum_j D_j (1 - r_j) \tag{13a}$$

$$\text{s.t.} \quad \frac{D_i r_j}{B_{ij}} \leq t - s \quad \forall (i,j) : i \neq j \tag{13b}$$

$$\sum_j r_j = 1 \tag{13c}$$

$$r_j \geq 0 \quad \forall j \tag{13d}$$

3.3 Optima

Since Problem (13) is a convex optimization problem (linear program), we look at the Karush-Kuhn-Tucker (KKT) conditions for optimality using the following dual variables $(\mu_{ij}, \theta, \lambda_j) : \forall (i,j), i \neq j$:

$$-D_j + \sum_{i \neq j} \frac{D_i}{B_{ij}} \mu_{ij} + \theta - \lambda_j = 0 \quad \forall j \tag{14a}$$

$$\mu_{ij} \left(\frac{D_i}{B_{ij}} r_j - (t - s) \right) = 0 \quad \forall (i,j) : i \neq j \tag{14b}$$

$$r_j \lambda_j = 0 \quad \forall j \tag{14c}$$

$$\mu_{ij} \geq 0 \quad \forall (i,j) : i \neq j \tag{14d}$$

$$\lambda_j \geq 0 \quad \forall j \tag{14e}$$

$$\frac{D_i}{B_{ij}} r_j - (t - s) \leq 0 \quad \forall (i,j) : i \neq j \tag{14f}$$

$$\sum_j r_j - 1 = 0 \tag{14g}$$

$$r_j \geq 0 \quad \forall j \tag{14h}$$

where (14a) are the first stationary conditions, (14b) (14c) are the complimentary slackness conditions, (14d) (14e) are the dual feasibility conditions, and (14f) (14g) (14h) are the primal feasibility conditions.

Notice that for every site j, (14f) implies that there could be bottlenecking link if $\max_{i \neq j} \{D_i / B_{ij}\} > t - s$. There would not be a bottlenecking link only if making $r_j = 1$ meets the deadline. Let us denote

$$U_j := \max_{i \neq j} \{D_i / B_{ij}\} \tag{15}$$

as the maximum upload time for site j if $r_j = 1$, and denote

$$b_j := \arg\max_{i \neq j} \{D_i / B_{ij}\} \tag{16}$$

as the set of bottlenecking data sources. If for any $i \notin b_j$, then (14f) is satisfied if it satisfied for b_j and also $D_i / B_{ij} < t - s$. This fact and (14b) implies that $\mu_{ij} = 0$ if $i \notin b_j$. Now the KKT conditions (14) have redundant conditions that can be eliminated to:

$$-D_j + U_j \sum_{i \in b_j} \mu_{ij} + \theta - \lambda_j = 0 \quad \forall j \tag{17a}$$

$$\mu_{ij} \left(U_j r_j - (t - s) \right) = 0 \quad \forall (i,j) : i \in b_j \tag{17b}$$

$$r_j \lambda_j = 0 \quad \forall j \tag{17c}$$

$$\mu_{ij} \geq 0 \quad \forall (i,j) : i \in b_j \tag{17d}$$

$$\lambda_j \geq 0 \quad \forall j \tag{17e}$$

$$U_j r_j - (t - s) \leq 0 \quad \forall (i,j) : i \in b_j \tag{17f}$$

$$\sum_j r_j - 1 = 0 \tag{17g}$$

$$r_j \geq 0 \quad \forall j \tag{17h}$$

We have three cases for each r_j:

(1) $r_j = (t - s)/U_j$. Since $t - s > 0$ and $U_j > 0$, then $r_j > 0$. Then (17c) results in $\lambda_j = 0$. In this case (17a) turns into:

$$\theta = D_j - U_j \sum_{i \in b_j} \mu_{ij} \tag{18}$$

Since $\sum_{i \in b_j} \mu_{ij} \geq 0$ from (17d), then this means that $\theta \leq D_j$.

(2) $0 < r_j < (t - s)/U_j$. Then (17c) results in $\lambda_j = 0$ and (17b) results in $\mu_{ij} = 0 : \forall i \in b_j$. In this case (17a) turns into:

$$\theta = D_j \tag{19}$$

(3) $r_j = 0$. Then (17b) results in $\mu_{ij} = 0 : \forall i \in b_j$. In this case (17a) turns into:

$$\theta = D_j + \lambda_j \tag{20}$$

Since (17e), then $\theta \geq D_j$.

Since θ can only be a single value, the above cases give a natural ordering. For a particular θ: if $D_j > \theta$, then Case 1 applies; if $D_j < \theta$ then Case 3 applies; if $D_j = \theta$ then Cases 1,2, or 3 could apply. This also means that we can restrict θ to the set $\{D_j : \forall j\}$ without restricting the solution space of the task placements $r_j : \forall j$.

Also, if all sites are in Case 1 and we observe that from (17g) $\sum_j \frac{1}{U_j} < \frac{1}{t-s}$, then the deadline t is too soon and the problem is infeasible. Let us denote the lower bound on t as:

$$\underline{t} := s + \frac{1}{\sum_j \frac{1}{U_j}} \tag{21}$$

On the other hand, let $l := \arg\max_j \{D_j\}$ and if $U_l < t - s$, then $r_l = 1$ and $r_j = 0 : \forall j \neq l$. Let us denote the upper bound on t as:

$$\bar{t} := s + U_l \tag{22}$$

If $t \geq \bar{t}$, then $r_l = 1$, $r_j = 0 : \forall j \neq l$, and the WAN usage is $\sum_i D_i - D_l$.

If $t = \underline{t}$, then $r_j = (\underline{t} - s)/U_j : \forall j$ and the WAN usage is $\sum_i D_i - (\underline{t} - s)\sum_j (D_j/U_j)$.

Note that if either the maximum deadline range

$$\bar{t} - \underline{t} = U_l - \frac{1}{\sum_j \frac{1}{U_j}} \tag{23}$$

$$\tag{24}$$

or the maximum WAN savings

$$D_l - \frac{1}{\sum_j \frac{1}{U_j}} \sum_j \frac{D_j}{U_j} \tag{25}$$

has significant cost, then developing an optimization algorithm is worthwhile.

3.4 Algorithm

We have the following algorithm:

(1) **Initialize:**
- Order D_j in descending order and relabel the indexes for this ordering.
- Set $y := 1$, $k := 1$, and $r_j := 0 : \forall j$
- For each site j, set:
  $U_j := \max_{i \neq j} \{D_i/B_{ij}\}$
  $b_j := \arg\max_{i \neq j} \{D_i/B_{ij}\}$.

(2) **Test for feasibility:**
- If $t \leq s + \frac{1}{\sum_j \frac{1}{U_j}}$, then stop because the problem is infeasible.

(3) **Process:**
- Replace $r_k := \min\{y, (t-s)/U_k\}$.
- Update $y := y - r_k$ and $k := k + 1$.
- If $y \leq 0$ or $k > |\mathcal{L}|$, then stop and output $r_j : \forall j$. Otherwise repeat Step (3).

APPENDIX