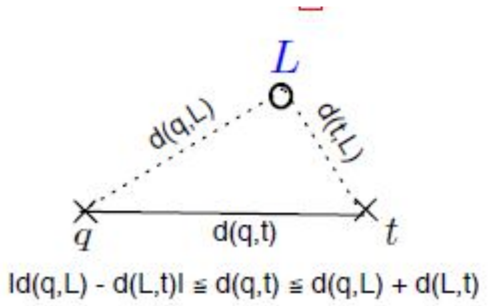
Generalizations of the Theory and Deployment of Triangular Inequality for Compiler-Based Strength Reduction

-- 李鹏辉 付琳晴 刘国栋

Background

- Traditional Strength Reduction --individual instruction/statement
  - Replacing expensive oprations with equivant but cheaper operations (e.g. 2xb => b << 1)
- Triangular inquality (ETI)



- Bound estimation has been widely deployed in many algorithms. e.g. k-nearest neighbors

k-nearest neighbors

- Distance bounds used to avoid unnecessary distance calculations

```
for i = 0 to N do
    minDist = Int_max;
    for j = 0 to M do
        dist = d(a(i), b(j));
        if minDist > dist
            minDist = dist;
            assign(i)= j;
```

(a)

```
for i = 0 to N do
    minDist = Int_max;
    for j = 0 to M do
        lbDist = lb(a(i), b(j));
        if minDist <= lbDist
          continue;
        ...

///lb() function for lower bound of distance
......
```

(b)

Motivation

- Distance computation among data points is essential in many algorithms of data analytics, graph analysis, machine learning and so on
- Traditional triangular inequality has been widely used in many manual algoritm designs

- Is ETI enough?
- Can integrate into compiler and automatically optimize deeply?

Contributions

- Develop a novel TI called Angle Triangular Inequality

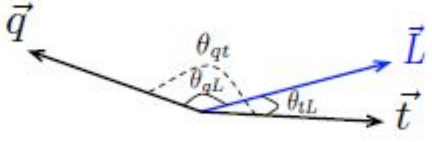
- Effectively deploy in TI optimizations using guided TI adaptation

- Intergrate the optimization technique into a LLVM based prototype compiler

Theorem 1. Angle Triangular Inequality

THEOREM 1. **Angle Triangular Inequality:** *For three arbitrary vectors $\vec{q}$, $\vec{t}$ and $\vec{L}$ in a space, the angles among them, denoted as θ_{qt}, θ_{qL}, θ_{tL}, must meet the following condition:*

$$cos(\theta_{qL} + \theta_{tL}) \leq cos\theta_{qt} \leq cos(\theta_{qL} - \theta_{tL}). \qquad (1)$$



Proof

Proof: Let $\vec{u_q}$, $\vec{u_t}$ and $\vec{u_L}$ represent three unit-length vectors in the direction of $\vec{q}$, $\vec{t}$ and $\vec{L}$ respectively.

We introduce two derived vectors

$$\vec{e_1} = \frac{\vec{u_q} - \vec{u_L} \cdot cos(\theta_{qL})}{sin(\theta_{qL})}$$

$$\vec{e_2} = \frac{\vec{u_t} - \vec{u_L} \cdot cos(\theta_{tL})}{sin(\theta_{tL})}.$$

Two unit vectors are perpendicular to $\vec{u_L}$

Proof - continued

$$\vec{u_q} = \vec{u_L} \cdot cos(\theta_{qL}) + \vec{e_1} \cdot sin(\theta_{qL})$$
$$\vec{u_t} = \vec{u_L} \cdot cos(\theta_{tL}) + \vec{e_2} \cdot sin(\theta_{tL}). \quad (2)$$

$$\vec{u_q} \cdot \vec{u_t} = cos(\theta_{qL})cos(\theta_{tL}) + \vec{e_1} \cdot \vec{e_2} sin(\theta_{qL})sin(\theta_{tL}).$$

Proof: Let $\vec{u_q}$, $\vec{u_t}$ and $\vec{u_L}$ represent three unit-length vectors in the direction of $\vec{q}$, $\vec{t}$ and $\vec{L}$ respectively.

We introduce two derived vectors

$$\vec{e_1} = \frac{\vec{u_q} - \vec{u_L} \cdot cos(\theta_{qL})}{sin(\theta_{qL})}$$

$$\vec{e_2} = \frac{\vec{u_t} - \vec{u_L} \cdot cos(\theta_{tL})}{sin(\theta_{tL})}.$$

$$\vec{u_q} \cdot \vec{u_t} \geq cos(\theta_{qL})cos(\theta_{tL}) - sin(\theta_{qL})sin(\theta_{tL})$$
$$\vec{u_q} \cdot \vec{u_t} \leq cos(\theta_{qL})cos(\theta_{tL}) + sin(\theta_{qL})sin(\theta_{tL}). \quad (3)$$

Proof - continued

Recall the *Trigonometric Addition Formulas*:

$$\begin{aligned} cos(\theta_1 + \theta_2) &= cos(\theta_1)cos(\theta_2) - sin(\theta_1)sin(\theta_1) \\ cos(\theta_1 - \theta_2) &= cos(\theta_1)cos(\theta_2) + sin(\theta_1)sin(\theta_1). \end{aligned} \tag{4}$$

Therefore, we have

$$cos(\theta_{qL} + \theta_{tL}) \leq \vec{u_q} \cdot \vec{u_t} \leq cos(\theta_{qL} - \theta_{tL}).$$

Because $\vec{u_q} \cdot \vec{u_t} = cos(\theta_{qt})$ as both u_q and u_t are unit vectors, we get

$$cos(\theta_{qL} + \theta_{tL}) \leq cos(\theta_{qt}) \leq cos(\theta_{qL} - \theta_{tL}).$$

The ATI theorem is hence proved. □

$$\begin{aligned} \vec{u_q} \cdot \vec{u_t} &\geq cos(\theta_{qL})cos(\theta_{tL}) - sin(\theta_{qL})sin(\theta_{tL}) \\ \vec{u_q} \cdot \vec{u_t} &\leq cos(\theta_{qL})cos(\theta_{tL}) + sin(\theta_{qL})sin(\theta_{tL}). \end{aligned} \tag{3}$$

- COROLLARY 1

COROLLARY 1. *For three arbitrary vectors $\vec{q}$, $\vec{t}$ and $\vec{L}$ in a space, the angles among them, denoted as θ_{qt}, θ_{qL}, θ_{tL}, must meet the following condition:*

$$|\theta_{qL} - \theta_{tL}| \leq \theta_{qt} \leq \pi - |\pi - (\theta_{qL} + \theta_{tL})|. \tag{5}$$

- COROLLARY 2

COROLLARY 2. *For three arbitrary vectors in a space $\vec{q}$, $\vec{t}$, $\vec{L}$, the following conditions must hold:*

$$\begin{aligned} \vec{q} \cdot \vec{t} &\geq |\vec{q}| \cdot |\vec{t}| \cdot cos(\theta_{qL} + \theta_{tL}) \\ \vec{q} \cdot \vec{t} &\leq |\vec{q}| \cdot |\vec{t}| \cdot cos(\theta_{qL} - \theta_{tL}|). \end{aligned} \tag{6}$$

THEOREM 1. **Angle Triangular Inequality:** *For three arbitrary vectors $\vec{q}$, $\vec{t}$ and $\vec{L}$ in a space, the angles among them, denoted as θ_{qt}, θ_{qL}, θ_{tL}, must meet the following condition:*

$$cos(\theta_{qL} + \theta_{tL}) \leq cos\theta_{qt} \leq cos(\theta_{qL} - \theta_{tL}). \tag{1}$$

Why ATI?

- Cosine values are used in computation between vectors
  - e.g. $\mathbf{A} \cdot \mathbf{B} = |\mathbf{A}||\mathbf{B}|\cos\langle \mathbf{A},\mathbf{B} \rangle$
  - e.g. $\mathbf{Distance}(\mathbf{A},\mathbf{B})^2 = |\mathbf{A}-\mathbf{B}|^2 = |\mathbf{A}|^2 + |\mathbf{B}|^2 - 2\mathbf{A} \cdot \mathbf{B}$

Theorem 2: Tighter ATI-based Distance Bound

- Traditional ETI bounds

$$
\begin{aligned}
|d(q,L) - d(t,L)| &\le \sqrt{|\vec{q}|^2 + |\vec{t}|^2 - 2|\vec{q}| \cdot |\vec{t}| \cdot cos(\theta_{qL} - \theta_{tL})} \\
d(q,L) + d(\vec{t},\vec{L}) &\ge \sqrt{|\vec{q}|^2 + |\vec{t}|^2 - 2|\vec{q}| \cdot |\vec{t}| \cdot cos(\theta_{qL} + \theta_{tL})}
\end{aligned} \tag{9}
$$

$$
\begin{aligned}
lb_{eti} =& \Big| \sqrt{|\vec{q}|^2 + |\vec{L}|^2 - 2|\vec{q}||\vec{L}|cos(\theta_{qL})} \\
& - \sqrt{|\vec{t}|^2 + |\vec{L}|^2 - 2|\vec{t}||\vec{L}|cos(\theta_{tL})} \Big| \\
ub_{eti} =& \sqrt{|\vec{q}|^2 + |\vec{L}|^2 - 2|\vec{q}||\vec{L}|cos(\theta_{qL})} \\
& + \sqrt{|\vec{t}|^2 + |\vec{L}|^2 - 2|\vec{t}||\vec{L}|cos(\theta_{tL})}.
\end{aligned} \tag{10}
$$

- To proof that for arbitrarily given $\mathbf{q}$,$\mathbf{t}$, and $\mathbf{L}$, the largest value of $\mathbf{lb}_{eti}$ is no larger than the lower bound of $d(q,t)$ given by ATI.

Proof

- When the condition for $\mathbf{lb}_{\mathbf{eti}}$ reaches maximal value when its derivative over $\mathbf{L}$ equals **0.**

$$\frac{d(lb_{eti})}{d|\vec{L}|} = 0.$$

Thus when
$$|\vec{L}| = \frac{|\vec{q}|\cdot|\vec{t}|\cdot sin(\theta_{tL}-\theta_{qL})}{|\vec{t}|\cdot sin(\theta_{tL})-|\vec{q}|\cdot sin(\theta_{qL}))}$$

$$\mathbf{lb}_{\mathbf{eti}} = \sqrt{|\vec{q}|^2 + |\vec{t}|^2 - 2|\vec{q}|\cdot|\vec{t}|\cdot cos(\theta_{qL} - \theta_{tL})}$$

COROLLARY 2. *For three arbitrary vectors in a space $\vec{q}$, $\vec{t}$, $\vec{L}$, the following conditions must hold:*

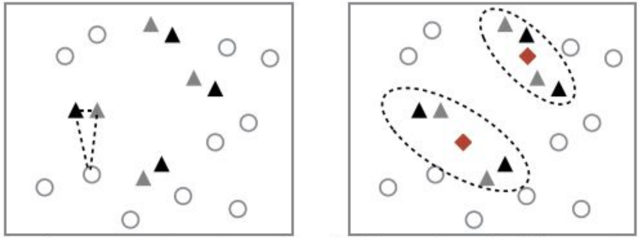
$$\begin{aligned}\vec{q}\cdot\vec{t} &\geq |\vec{q}|\cdot|\vec{t}|\cdot cos(\theta_{qL}+\theta_{tL})\\ \vec{q}\cdot\vec{t} &\leq |\vec{q}|\cdot|\vec{t}|\cdot cos(\theta_{qL}-\theta_{tL}|).\end{aligned} \tag{6}$$

Similar for right hand side bounds.

Select landmark

- Previous work:

(1) No-iterative algorithm: select in lightweight clustering.
(2) Iterative algorithm: use previous iteration as landmarks.
(3) Stringent memory space: use group filtering



(c) ghosts as landmarks (d) landmark hierarchy

- Problems:

(1) All based on ETI
(2) Ambiguous (Such as: dimension is not large)

ETI & ATI in Group Filtering

Distance calculation:

ATI:

$$lb(d(q,G)) = \sqrt{|\vec{q}|^2 + \min_{\vec{t} \in G}(|\vec{t}|^2 - 2|\vec{q}| \cdot |\vec{t}| \cdot cos(\theta_{qL} - \theta_{tL}))};$$
$$ub(d(q,G)) = \sqrt{|\vec{q}|^2 + \max_{\vec{t} \in G}(|\vec{t}|^2 - 2|\vec{q}| \cdot |\vec{t}| \cdot cos(\theta_{qL} + \theta_{tL}))}. \tag{16}$$

ETI:

$$lb(d(q,G)) = d(q,L) - \max_{\vec{t} \in G} d(L,t);$$
$$ub(d(q,G)) = d(q,L) + \max_{\vec{t} \in G} d(L,t). \tag{17}$$

√

ETI & ATI in Group Filtering

Cosine Similarity:

THEOREM 1. **Angle Triangular Inequality:** *For three arbitrary vectors $\vec{q}$, $\vec{t}$ and $\vec{L}$ in a space, the angles among them, denoted as θ_{qt}, θ_{qL}, θ_{tL}, must meet the following condition:*

$$cos(\theta_{qL} + \theta_{tL}) \leq cos\theta_{qt} \leq cos(\theta_{qL} - \theta_{tL}). \qquad (1)$$

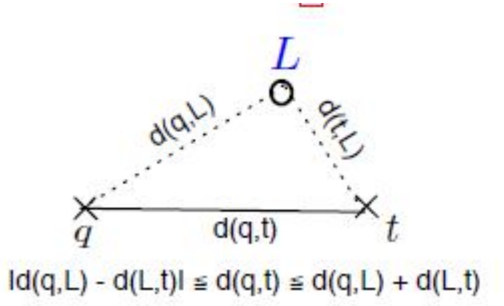
ATI :

$$lb(cos(\theta_{\vec{q},G})) = \begin{cases} cos(\theta_{\vec{q},\vec{L}} + \max_{\vec{t} \in G} \theta_{\vec{L},\vec{t}}) & \text{if } \theta_{\vec{q},\vec{L}} + \max_{\vec{t} \in G} \theta_{\vec{L},\vec{t}} \leq \pi; \\ -1 & \text{otherwise.} \end{cases}$$

$$ub(cos(\theta_{\vec{q},G})) = \begin{cases} cos(\theta_{\vec{q},\vec{L}} - \max_{\vec{t} \in G} \theta_{\vec{L},\vec{t}}) & \text{if } \max_{\vec{t} \in G} \theta_{\vec{q},\vec{t}} \leq \theta_{\vec{q},\vec{L}}; \\ 1 & \text{otherwise.} \end{cases} \qquad (18)$$

ETI & ATI in Early Stop

Check whether distances between point q and points in G are smaller than C:



Sort points in a descending order of d(t, L), check whether d(q, L) + d(t, L)(the upper bound) is smaller than C

ETI : √

Insights:

ATI should be used alone when optimizing Cosine Similarity.

ATI should be combined with ETI when optimizing distance calculation.

Algorithm combined ATI & ETI in distance calculation

```
//check whether group filtering is applicable
....

if (group filtering is applicable)
   //prepare for group-level filtering with ETI
   for L in Landmarks do
      sort target points in L based on their distances to L

for i = 0 to |Q| do
   //ETI for group-level filtering
   for L in Landmarks do
      if ETI_bound(Q[i], L) passes the comparison
         continue;
      for target point t in L do
         //ETI for point-level filtering
         if ETI_bound(Q[i], t) passes the comparison
            break;
         //ATI for point-level filtering
         if ATI_bound(Q[i], t) passes the comparison
            continue;
         //if all previous filtering fails, run the original code.
         .....
```

Guided TI Adaptation: determine suitable configuration

- Classify the program into: **non-iterative & iterative** distance calculation or cosine similarity
- Use built-in performance model to calculate the time saving

$$
\begin{aligned}
T_{save} &= T_{savedDistance} - T_{overhead}; \\
\boxed{T_{savedDistance} = (r_d \cdot n \cdot m) \cdot t_{distance};} \\
T_{overhead} &= T_{createLM} + T_{LMdistance} + T_{checks} \\
&\simeq (p \cdot m \cdot k) \cdot t_{distance} \\
&+ (n + m) \cdot t_{distance} \\
&+ (r_c \cdot n \cdot m + n \cdot k) \cdot t_{checks};
\end{aligned} \tag{20}
$$

k: number of landmarks

- Form a small sample and find the best number of landmarks

Integration with Compilers

- Through Pattern Matching
- Through Assistance of API

Integration with Compilers

- Through Pattern Matching
  - avoid computation of some of the results.
  - allowed usage patterns

```
p = a calculated distance or dot product
q = f (p); // some value is derived from p;
           // f() is a monotonic relation;
           // inside f(), x is indep. of p;

if (q op x){ // op is a comparison operator
  body_1;
} // body_1 may read or write p or q
body_2; // no reads of p or q
```

(a) Pattern 1

```
p = a calculated distance or dot product
q = f (p); // some value is derived from p;
           // f() is a monotonic relation;
           // inside f(), x is indep. of p;

if (q op x){ // op is a comparison operator
  body_3;
}
else{
  body_4;
} // only one of body_3 and body_4 can read p or q
body_5; // no reads of p or q
```

(b) Pattern 2

Figure 7. Allowed usage patterns of distances or dot products.

Integration with Compilers

- Through Assistance of API

```
_SR_dotProduct(_SR_vector, _SR_vector);
_SR_vectorMatrixProduct(_SR_vector, _SR_matrix);
_SR_mm(_SR_matrix, _SR_matrix);
_SR_defDistance (enum);
_SR_getLowerBound (_SR_pointSet, _SR_pointSet);
_SR_getUpperBound (_SR_pointSet, _SR_pointSet);
_SR_findClosestTargets (int, _SR_pointSet, _SR_pointSet);
_SR_findFarthestTargets (int, _SR_pointSet, _SR_pointSet);
_SR_findTargetsWithin (float, _SR_pointSet, _SR_pointSet);
_SR_findTargetsBeyond (float, _SR_pointSet, _SR_pointSet);
_SR_update (_SR_pointSet, ...);
```

Figure 8. Core APIs for assisting TI-based Strength Reduction.

Evaluation

- 8 influential algorithms from various domains
  - KNNJoin
  - KNN
  - KMeans
  - ICP
  - Nbody
  - DC
  - KDR
  - RBM

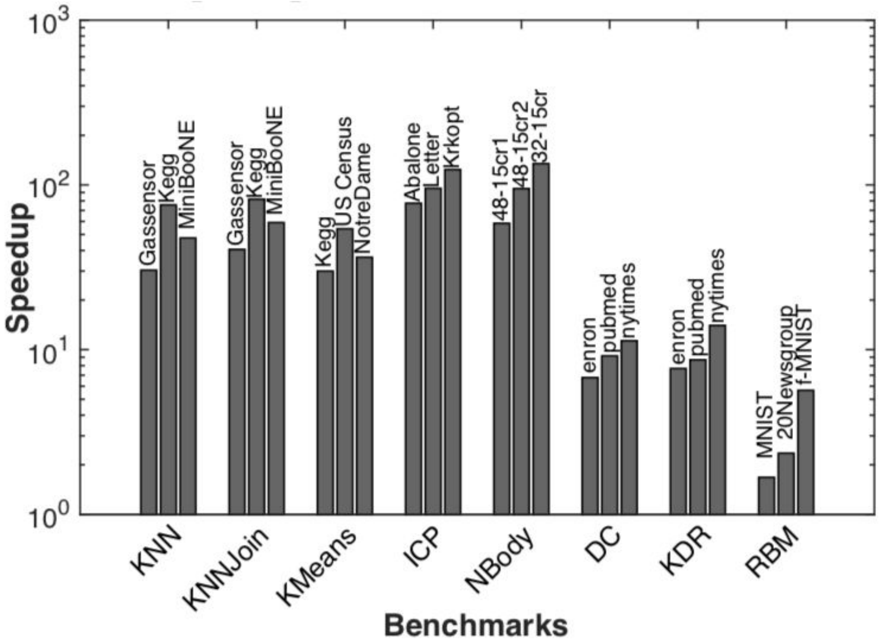
distance computation

dot products computation

- compared with 2 other versions
  - standard
  - optimized (TOP)

Evaluation

- Speedup over standard version



- Achieves as much as 134X(Nbody) and 46X on average.
- Over 91% computation savings for all the datasets.(Except RBM).
- At least 93% and frequently over 99% of the distance computations
- At least 91% and frequently over 94% of the vector product computations

Evaluation

- Speedup over optimized version

Prog	KNN	KNNjoin	KMeans	ICP	Nbody	DC	KDR	RBM	*geomean*
Speedup	1.35X	1.46X	1.19X	1.17X	1.14X	9.19X	10.13X	3.23X	2.35

  - The extra speedup comes from 2 aspects
    - ATI
    - guied TI adaption

Evaluation

- Tighter bounds by ATI

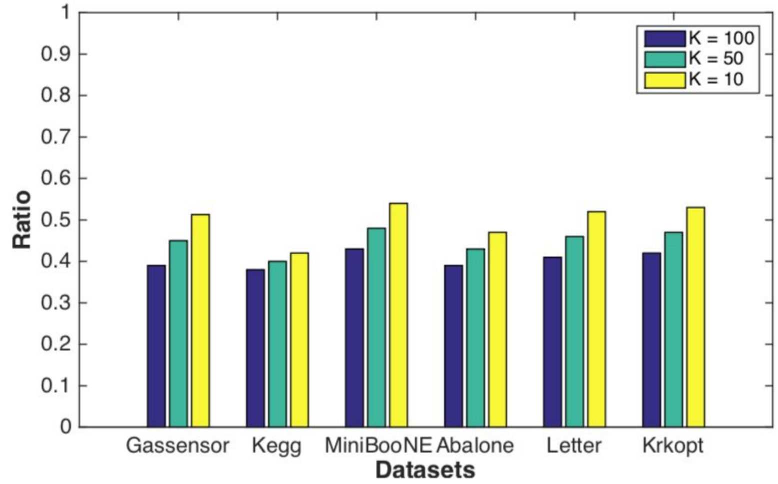


Figure 11. Fraction of extra savings of the distance computations due to the tighter bounds by ATI over those by ETI on KNNJoin.

Evaluation

- Guided TI Adaption

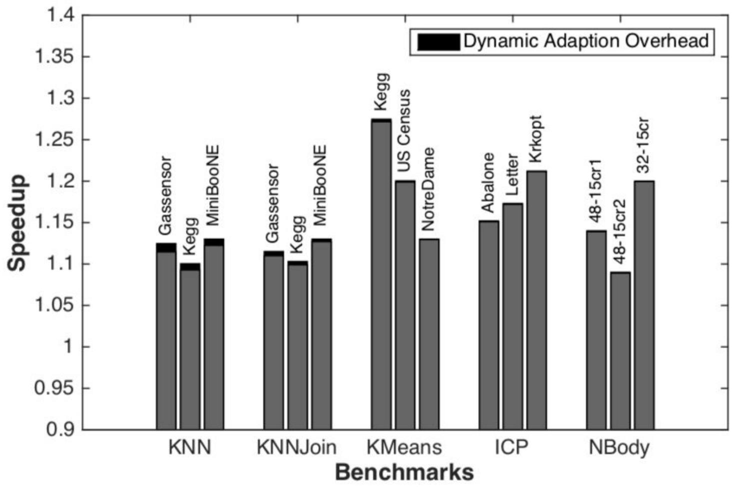


Figure 10. Speedup brought by the guided TI adaptation over using rigid rules [15] for deploying TI-based optimizations. The top black segment on each bar represents the overhead incurred by the runtime sampling and adaptation.

Related Work

- This work was inspired by TOP, but makes some significant extensions in both theory and implementation.
- This paper is the first that proposes the concept of TI-based strength reduction.
- Removing redundant computations from a program is a classic topic in compiler.
- Triangular inequality has been used in the design of many algorithms.
  - All these are manual algorithm designs, and exploit only ETI.
- Recent years witnessed some development of approximation-based program optimizations, TI-based strength reduction uses no approximations, and hence introduces no errors into the computation results.