Speeding Up Dynamic Programming via Quadrangle Inequality

Pramook Khungurn

March 14, 2013

1 Maximum t-gon Problem

- We are given a set of points P in $\mathbb{R}^2$, and you are to find a convex polygon with no more than t vertices whose vertices are from P such that its perimeter is the longest.
- It is easy to show that we only need to concern ourselves with the vertices of the convex hull of these points. So let the convex hull have n vertices, and let them be called $v_1, v_2, \ldots, v_n$ in clockwise order.
- Solving the maxinum t-gon problem can be overwhelming. How about solving the maximum *triangle* problem first?
- Let D be an $n \times n$ matrix such that the element in the ith row and the jth column, d_{ij}, equals the Euclidian distance between v_i and v_j.
- Let $D \otimes D$ denote another $n \times n$ matrix where
$$(D \otimes D)_{ij} = \max_{i \le k \le j} \{d_{ik} + d_{kj}\}.$$
That is, $(D \otimes D)_{ij}$ is the longest convex two-hops-or-less path from v_i to v_j.
- It follows that the solution to the maximum triangle problem lies in the maximum element of the matrix $D + (D \otimes D)$. Hence, by the standard dynamic programming technique, we can compute the matrix and its maximum element in $O(n^3)$ times.
- Of course, we can do better than this by exploiting a special structure of the matrix. Let $K(i,j)$ denote the maximum intenger k such that $d_{ik} + d_{kj} = (D \otimes D)_{ij}$. In other words,
$$K(i,j) = \max\{k \mid d_{ik} + d_{kj} = (D \otimes D)_{ij}\}.$$
We can show that the following claim is true.

Claim 1. *K is* monotone. *That is,*
$$K(i,j) \le K(i,j+1) \le K(i+1,j+1).$$
- By virtue of the claim, we can compute $D \otimes D$ faster by computing the elements in the order of increasing $j - i$. Now, when we want to compute $(D \otimes D)_{ij}$, we vary k from $K(i,j-1)$ to $K(i+1,j)$ instead of varying it from i to j like we used to do.

The pseudocode for computing $D \otimes D$ (stored in table M) and K (stored in the eponymous table) is given in Algorithm 1.
- What is the time complexity of Algorithm 1? For each value of δ, the innermost loop runs
$$\sum_{i=1}^{n-\delta-1} \Big[K(i+1,i+1+\delta) - K(i,i+\delta) + 1\Big] = K(n-\delta,n) - K(1,1+\delta) + n - \delta - 1 \le 2n$$
times. Since δ takes value from 1 to $n-1$, we have that Algorithm 1 has running time $O(n^2)$.

Algorithm 1 Computing $D \otimes D$

```
for i = 1 to n do
  K[i, i] = i
  M[i, i] = 0
end for
for δ = 1 to n − 1 do
  for i = 1 to n − δ do
    j = i + δ
    M[i, j] = −∞
    for k = K[i, j − 1] to K[i + 1, j] do
      if d_ik + d_kj ≥ M[i, j] then
        M[i, j] = d_ik + d_kj
        K[i, j] = k
      end if
    end for
  end for
end for
```

2 Quadrangle Inequality

- The key property of the matrix D that makes K monotone is that it satisfies the *quadrangle inequality* (QI): for all $i \le i' \le j \le j'$,
$$d_{ij} + d_{i'j'} \ge d_{ij'} + d_{i'j}.$$
This statement can be proven using elementary geometry.

- In this section, we discuss properties of 2D functions satisfying the quadrangle inequality. In particular, we shall discuss how the quadrangle inequality gives rise to monotonicity of K above. Before we discuss the proof, however, some definitions are in order.

Definition 2. *A function $f(\cdot,\cdot)$ is said to satisfy the* convex quadrangle inequality *if, for all $i \le i' \le j \le j'$, we have*
$$f(i,j) + f(i',j') \ge f(i,j') + f(i',j).$$
We say that it satisfies the concave quadrangle inequality *if, for all $i \le i' \le j \le j'$, we have*
$$f(i,j) + f(i',j') \le f(i,j') + f(i',j).$$

Definition 3. *Let $f(\cdot,\cdot)$ and $g(\cdot,\cdot)$ be 2D functions. Then, $f \otimes g$ is another 2D function, where*
$$(f \otimes g)(i,j) = \max_{i \le k \le j} \{f(i,k) + g(k,j)\}.$$
And $f \oslash g$ is another 2D function, where
$$(f \oslash g)(i,j) = \min_{i \le k \le j} \{f(i,k) + g(k,j)\}.$$

- We now prove an important property of $\otimes$ and $\oslash$.

Lemma 4. *If both f and g satisfies convex QI, then so does $f \otimes g$. On the other hand, if both f and g satisfies concave QI, then so does $f \oslash g$.*

Proof. We shall only work on the convex case as the concave case can be proven with the same reasoning.

Suppose that both f and g satisfy convex QI. Let $h = f \oplus g$. Let i, i', j, and j' be integers such that $i \le i' \le j \le j'$

Let y be integer such that $h(i,j') = f(i,y) + g(y,j')$. Let z be integer such that $h(i',j) = f(i',z) = g(z,j)$. For convenience, let us denote $f(i,k) + g(k,j)$ with $h_k(i,j)$, so that we may write $h(i,j') = h_k(i,j')$ and $h(i',j) = h_z(i',j)$. There are two cases to consider.

In the first case, $y \le z$. We can deduce that $y \le j$. Now,

$$\begin{aligned}
h(i,j) + h(i',j') &\ge h_y(i,j) + h_z(i',j') \\
&\ge f(i,y) + g(y,j) + f(i',z) + f(z,j') \\
&= f(i,y) + f(i',z) + [g(y,j) + g(z,j')] \\
&\ge f(i,y) + f(i',z) + [g(y,j') + g(z,j)] \\
&= [f(i,y) + g(y,j')] + [f(i',z) + g(z,j)] \\
&= h_y(i,j') + h_z(i',j) \\
&= h(i,j') + h(i',j)
\end{aligned}$$

The second case, where $y \ge z$, is symmetric to the first case. We are done □

- We now show that convex/concave QI implies monotonicity.

Lemma 5. *Let $h = f \otimes g$, and let $K(i,j) = \max\{k \mid h(i,j) = h_k(i,j)\}$. If f and g satisfy convex QI, then K is monotone; that is,*

$$K(i,j) \le K(i,j+1) \le K(i+1,j+1)$$

for all $i \le j$.

The same goes for $h = f \oslash g$ if both f and g satisfy concave QI.

Proof. Again, we work out only the convex case.

We shall prove the first inequality, $K(i,j) \le K(i,j+1)$, by showing that, for any k and k' such that $i \le k \le k' \le j$, we have that

$$[h_{k'}(i,j) \ge h_k(i,j)] \implies [h_{k'}(i,j+1) \ge h_k(i,j+1)]. \quad (1)$$

Applying the convex QI g with $k \le k' \le j \le j+1$, we have

$$g(k,j) + g(k',j+1) \ge g(k,j+1) + g(k',j)$$

Adding $f(i,k) + f(i,k')$ to both sides, we have that

$$\begin{gathered}
[f(i,k) + g(k,j)] + [f(i,k') + g(k',j+1)] \ge [f(i,k) + g(k,j+1)] + [f(i,k') + g(k',j)] \\
h_k(i,j) + h_{k'}(i,j+1) \ge h_k(i,j+1) + h_{k'}(i,j).
\end{gathered}$$

which implies (1). The second inequality, $K(i,j+1) \le K(i+1,j+1)$, is symmetric and can be proven similarly. □

Theorem 6. *If both f and g satisfy convex QI, then $f \otimes g$ can be computed in $O(n^2)$ time. Also, if both f and g satisfy concave QI, then $f \oslash g$ can be computed in $O(n^2)$ time as well.*

3 Maximum t-gon Problem (Revisited)

- To compute the solution to the problem efficiently, we make use of the following property of $\otimes$.

  Lemma 7. *$\otimes$ is associative.*

  Proof. Let f, g, and h be 2D functions. We have that

$$\begin{aligned}[f \otimes (g \otimes h)](i,j) &= \max_{i \le k \le j} \{f(i,k) + (g \otimes h)(k,j)\} \\ &= \max_{i \le k \le j} \left\{ f(i,k) + \max_{k \le \ell \le j} \{g(k,\ell) + h(\ell,j)\} \right\} \\ &= \max_{i \le k \le \ell \le j} \{f(i,k) + g(k,\ell) + h(\ell,j)\}.\end{aligned}$$

  We can similarly show that $[(f \otimes g) \otimes h](i,j)$ equals the same thing. □

- Hence, we can write $D \otimes D \otimes D$ without parentheses.

  For notational convenience, let us write $D \otimes D$ as $D^{(2)}$, $D \otimes D \otimes D$ as $D^{(3)}$, and so on.

- By Lemma 4, we have that $D^{(m)}$ satisfies convex QI for all $m \ge 1$. Hence, $D^{(m)}$ can be computed in $O(n^2 \log m)$ by deploying recursive squaring algorithm and using an algorithm similar to Algorihtm 1 to do the squaring.

- The solution to the maximum t-gon problem is the maximum entry of $D + D^{(t)}$. Thus, the problem can be solved in $O(n^2 \log t)$ time.

4 String Product Problem

- Let A and B be sets of strings. Let AB denote the set of *concatenations* of elements of A by elements of B. That is,

$$AB = \{ab \mid a \in A \wedge b \in B\}.$$

- You have a set $L_1, L_2, \dots, L_n$ of strings. You wish to compute the concatenation set $L_1 L_2 \cdots L_n$. However, the computation is restricted to concatenating two sets at a time. That is, you have to parenthesize the concatenation $L_1 L_2 \cdots L_n$ and carry out the binary concatenation as parenthesized.

  Let the cost of computing AB be $|A||B|$, the size of the output set. You wish to compute the parenthesization with the lowest cost.

- Let ℓ_i denote $|L_i|$, and let $w(i,j) = \ell_i \ell_{i+1} \cdots \ell_j$. Let $c(i,j)$ be the minimum cost of computing $L_i L_{i+1} \cdots L_j$. It is clear that $c(i,j)$ can be computed by the following recurrence relation:

$$c(i,j) = \begin{cases} 0, & i = j \\ w(i,j) + \min_{i < k \le j}\{c(i,k-1) + c(k,j)\}, & i < j \end{cases}$$

  Hence, the problem can be solve in $O(n^3)$ by standard dynamic programming.

- It turns out that we can compute 2D functions defined with recurrences of the form above in $O(n^2)$ if w satisfies concave QI, and if w is *monotone*.

  Definition 8. *A 2D function $w(\cdot,\cdot)$ is said to be* monotone *if, for all $i' \le i \le j \le j'$,*

$$w(i,j) \le w(i',j').$$

- Before proving the above result, let us demonstrate that $w(i,j) = \ell_i \ell_{i+1} \cdots \ell_j$ satisfies concave QI.

  Let $i \le i' \le j \le j'$. Let $a = w(i, i'-1)$, $b = w(i', j)$, and $c = w(j+1, j')$. Here, we abuse the notation and say that $w(i,j) = 1$ if $i > j$. We have that

  $$w(i,j) + w(i',j') = ab + bc = b(a+c) \le b(ac+1) \le abc + b = w(i,j') + w(i',j)$$

  where the fact that $a + c \le ac + 1$ comes from the fact that $(a-1)(c-1) \ge 0$. (Each set has to have at least an element. Otherwise, we wouldn't bother computing c in the first place.)

  It is also evident that $w(i,j)$ is monotone because $\ell_i \ge 1$ for all i.

- The proof that we can compute c as defined in (4) in $O(n^2)$ is very similar to the proof of Theorem 6. We first show that c also satisfies concave QI, and then show that the maximum index function K is monotone in the sense of Claim 1.

  Lemma 9. *Let c be defined as in (4), and let w satisfy concave QI and be monotone in the sense of Definition 8. Then, c also satisfies concave QI.*

  Proof. The proof is by induction on $j' - i$.

  As the base case, $i = i' = j = j'$, QI is trivially true.

  Now, suppose that QI holds for all non-negative $j' - i$ up to δ. Now, let $j' - i = \delta + 1$. There are two cases to consider.

  Case A: $i' = j$. In this case, we have to show that $c(i,j) + c(j,j') \le c(i,j')$. Now, let k be an integer such that $w(i,j') = w(i,j') + c(i,k-1) + c(k,j')$. There are two cases to consider regarding the value of k.

  - $k \le j$. In this case, we have that

    $$\begin{aligned} c(i,j) + c(j,j') &\le c_k(i,j) + c(j,j') \\ &= w(i,j) + c(i,k-1) + c(k,j) + c(j,j') \\ &\le w(i,j') + c(i,k-1) + c(k,j') \\ &= c(i,j') \end{aligned}$$

    where the identity $c(k,j) + c(j,j') \le c(k,j')$ is derived from the induction hypothesis.
  - $k > j$. This case is symmetric to the last case. We shall omit the proof.

  Case B: $i' < j$. Let $c_k(i,j) = w(i,j) + c(i,k-1) + c(k,j)$. Let y be an integer such that $c(i,j') = c_y(i,j')$, and let z be an integer such that $c(i',j) = c_z(i',j)$. There are again two cases to consider.

  - $y \le z$. We have that $i < y \le z \le j$. Hence,

    $$\begin{aligned} c(i,j) + c(i',j') &\le c_y(i,k) + c_z(i',j') \\ &\le w(i,j) + c(i,y-1) + c(y,j) + w(i',j') + c(i',z-1) + c(z,j') \\ &= [w(i,j) + w(i',j')] + c(i,y-1) + c(i',z-1) + [c(y,j) + c(z,j')] \\ &\le [w(i,j') + w(i',j)] + c(i,y-1) + c(i',z-1) + [c(y,j') + c(z,j)] \\ &= [w(i,j') + c(i,y-1) + c(y,j')] + [w(i',j) + c(i',z-1) + c(z,j)] \\ &= c_y(i,j') + c_z(i',j) \\ &= c(i,j') + c(i',j). \end{aligned}$$

  - $y > z$. This case is symmetric to the last case. We shall omit the proof (again).

  By induction, QI holds for all value of $j' - i$. $\square$

Lemma 10. *Let c be defined as in (4), and let w be monotone and satisfy concave QI. Let $K(i,j) = \max\{k \mid c(i,j) = c_k(i,j)\}$. Then, $K(i,j+1) \le K(i,j) \le K(i+1,j+1)$ for all $i \le j$.*

Proof. Similar to the proof of Lemma 5, we prove that the first inequality, $K(i,j) \le K(i+1,j+1)$, by showing that the following proposition is true: for all $k' > k$,

$$[c_{k'}(i,j) \le c_k(i,j)] \implies [c_{k'}(i,j+1) \le c_k(i,j+1)]. \tag{2}$$

Let k and k' be integers such that $k < k' \le j \le j+1$. Applying concave QI, we have that

$$c(k,j) + c(k',j+1) \le c(k,j+1) + c(k',j).$$

Adding $w(i,j) + w(i,j+1) + c(i,k-1) + c(i,k'-1)$ to both sides, we have

$$c_k(i,j) + c_{k'}(i,j+1) \le c_k(i,j+1) + c_{k'}(k',j)$$

which implies (2).

The second inequality can be proven in the same way. □

Theorem 11. *Let c be defined as in (4), and let w be monotone and satisfy concave QI. Then, c can be computed in $O(n^2)$ time.*

- We have been only discussing the case where $c(i,i) = 0$ for all i. However, $c(i,i)$ can take any value a_i because, then, $c(i,j) = c_{\text{old}}(i,j) + \sum_{k=i}^{j} a_k$, and QI still holds.

  Corollary 12. *Let c be defined as follows:*

  $$c(i,j) = \begin{cases} a_i, & i = j \\ w(i,j) + \min_{i<k\le j}\{c(i,k-1) + c(k,j)\}, & i < j \end{cases}$$

  for any constant $a_1, a_2, \ldots, a_n$. Then, if w satisfies concave QI and is monotone in the sense of Definition 8, then c satisfies QI, K is monotone in the sense of Claim 1, and thus c can be computed in $O(n^2)$.

- One famous example of dynamic programming problems that can be sped up this way is the optimal binary search tree problem. You are given a set of n distinct English words and you are to build an optimal binary search tree that minimize the expected search time according to these specifications:

  1. The ith word has the probability of being searched p_i.
  2. The probability of the search falling between the ith and the $(i+1)$st word is q_i. (So we have q_0, q_1, q_2, $\ldots$, q_n)
  3. Words are internal nodes, and leaves represent words that fall between them. (So there are $2n+1$ nodes in total.)
  4. A node at depth d incur a search time of d.

  Let $c(i,j)$ be the cost of the minimum binary search tree for words from the $(i+1)$st word to the jth word (i.e., the binary tree would consist of nodes for q_i, p_{i+1}, q_{i+1}, $\ldots$, p_j, and q_j). Then, $c(i,j)$ can be defined as in (4) with $w(i,j) = q_i + p_{i+1} + q_{i+1} + \cdots + p_j + q_j$, which can be shown easily to satisfy concave QI and to be monotone.

  Hence, the optimal binary search tree problem can be solved in $O(n^2)$ time.

5 Optimal t-ary Tree Problem

- What if, instead of constructing a binary search tree, we want to construct a t-ary search tree? If we restrict ourselves to the case that every search request must hit a word, the situation may be viewed as trying to compute c where

$$c(i,j) = \begin{cases} 0, & i \geq j \\ w(i,j) + \min_{i<k_1\leq k_2 \leq \cdots \leq k_{t-1} \leq j}, \{c(i,k_1-1) + c(k_1,k_2-1) + \cdots + c(k_{t-1},j)\}, & i < j \end{cases} \quad (3)$$

  where $w(i,j) = p_{i+1} + p_{i+2} + \cdots + p_j$.

- The main result in this section is that it is possible to compute c in $O(n^2 \log t)$ time if w is non-negative and satisfies convex QI and the *triangle inequality* (TI):

  Definition 13. *Function $w(\cdot,\cdot)$ is said to satisfy the* triangle inequality *if, for all $i < j < k$, we have*

$$w(i,j) + w(j,k) \leq w(i,k).$$

  Note that TI and w's non-negativity implies monotonicity in the sense of Definition 8.

- To simplify the notation, we define

$$f^{(t)}(i,j) = \begin{cases} 0, & i \geq j \\ \min_{i<k_1\leq k_2 \leq \cdots \leq k_{t-1} \leq j}, \{c(i,k_1-1) + c(k_1,k_2-1) + \cdots + c(k_{t-1},j)\}, & i < j \end{cases}$$

  so that we can write $c(i,j) = w(i,j) + f^{(t)}(i,j)$.

  Furthermore, for any q such that $1 \leq q < t$, we define

$$f^{(q)}(i,j) = \begin{cases} 0, & i \geq j \\ \min_{i\leq k_1\leq k_2 \leq \cdots \leq k_{q-1} \leq j}, \{c(i,k_1-1) + c(k_1,k_2-1) + \cdots + c(k_{q-1},j)\}, & i < j \end{cases}$$

  Note that $f^{(t)}(i,j)$ and $f^{(q)}(i,j)$ differs only at whether k_1 being strictly greater than i or not. $f^{(t)}(i,j)$ requires that the interval $[i,j]$ be split, but $f^{(q)}(i,j)$ do not.

  Note also that $f^{(1)}(i,j) = c$.

- We now state and give brief proofs of some properties of $f^{(t)}(i,j)$ and $f^{(q)}(i,j)$.

  Proposition 14. $f^{(1)}(i,j) \geq f^{(2)}(i,j) \geq \cdots \geq f^{(t-1)}(i,j) \geq f^{(t)}(i,j)$

  Proof. The inequealities hold by definition of $f^{(q)}(i,j)$ except for the last one: $f^{(t-1)}(i,j) \geq f^{(t)}(i,j)$. Now, if $f^{(t-1)}(i,j)$ achieves its minimum in such a way that the interval $[i,j]$ is split, then $f^{(t-1)}(i,j) \geq f^{(t)}(i,j)$. Otherwise, $f^{(t-1)}(i,j) = c(i,j) \geq f^{(t)}(i,j)$ because $w(i,j) \geq 0$. □

  Proposition 15. *For any q such that $2 \leq q < t$, and for any r and s such that $r \geq 1$, $s \geq 1$, and $r + s = q$, we have that*

$$f^{(q)}(i,j) = \min_{i\leq k \leq j} \{f^{(r)}(i,k-1) + f^{(s)}(k,j)\}$$

  Proof. It is evident that LHS $\leq$ RHS. To show that LHS $\geq$ RHS, suppose that the value of $f^{(q)}(i,j)$ is achieved by using $k_1, k_2, \ldots, k_r, \ldots, k_{q-1}$. Then, $f^{(q)}(i,j) \geq f^{(r)}(i,k_{r-1}) + f^{(s)}(k_r,j) \geq$ RHS. □

  Proposition 16. *For any r and s such that $r \geq 1$, $s \geq 1$, $t = r + s$, we have*

$$f^{(t)}(i,j) = \min_{i<k \leq j} \{f^{(r)}(i,k-1) + f^{(s)}(k,j)\}$$

Proof. Same as that of the proof of Proposition 15. ☐

- We now prove that $f^{(q)}$ and $f^{(t)}$ satisfy concave QI.

Lemma 17. *Let q be an integer such that $2 \leq q < t$. Let r and s be integers such that $r \geq 1$, $s \geq 1$, and $r+s=q$. If $f^{(r)}(i,j)$ and $f^{(s)}(i,j)$ satisfy concave QI for $j-i \leq \delta$, then $f^{(q)}(i,j)$ satisfies concave QI for $j-i \leq \delta$ as well.*

Proof. Let i, i', j, j' be integers such that $i \leq i' \leq j \leq j'$ with $j' - i \leq \delta$. There are two cases to consider.

Case A: $i' = j$. We have to show that $f^{(q)}(i,j) + f^{(q)}(j,j') \leq f^{(q)}(i,j')$. Let k be the integer such that $f^{(q)}(i,j') = f^{(r)}(i,k-1) + f^{(s)}(k,j')$. There two cases to consider regarding the value of k.

– $k \leq j$. Then, we have that

$$\begin{aligned}
f^{(q)}(i,j) + f^{(q)}(j,j') &\leq f^{(r)}(i,k-1) + f^{(s)}(k,j) + f^{(q)}(j,j') \\
&\leq f^{(r)}(i,k-1) + f^{(s)}(k,j) + f^{(s)}(j,j') \\
&\leq f^{(r)}(i,k-1) + f^{(s)}(k,j') + f^{(s)}(j,j) \\
&= f^{(r)}(i,k-1) + f^{(s)}(k,j') \\
&= f^{(q)}(i,j')
\end{aligned}$$

– $k > j$. This case is symmetric to the last case, and we will omit the proof.

Case B: $i' < j$. Let y be the integer such that $f^{(q)}(i,j') = f^{(r)}(i,y-1) + f^{(s)}(y,j')$, and z be the integer such that $f^{(q)}(i',j) = f^{(r)}(i',z-1) + f^{(s)}(z,j)$. There are two cases to consider.

– $y \leq z$. We have that $i \leq y \leq z \leq j$. So,

$$\begin{aligned}
f^{(q)}(i,j) + f^{(q)}(i',j') &\leq f^{(r)}(i,y-1) + f^{(s)}(y,j) + f^{(r)}(i',z-1) + f^{(s)}(z,j') \\
&= f^{(r)}(i,y-1) + f^{(r)}(i',z-1) + [f^{(s)}(y,j) + f^{(s)}(z,j')] \\
&\leq f^{(r)}(i,y-1) + f^{(r)}(i',z-1) + [f^{(s)}(y,j') + f^{(s)}(z,j)] \\
&= [f^{(r)}(i,y-1) + f^{(s)}(y,j')] + [f^{(r)}(i',z-1) + f^{(s)}(z,j)] \\
&= f^{(q)}(i,j') + f^{(q)}(i',j)
\end{aligned}$$

– $y > z$. This case is symmetric to the last case, and we will omit the proof.

We can now conclude that $f^{(q)}$ satisfies concave QI as well. ☐

Lemma 18. *Let r and s be integers such that $r \geq 1$, $s \geq 1$, and $r+s=t$. If $f^{(r)}(i,j)$ and $f^{(s)}(i,j)$ satisfy concave QI for $j-i \leq \delta$, then $f^{(t)}(i,j)$ satisfies concave QI for $j-i \leq \delta+1$.*

Proof. Pretty much the same as the proof of Lemma 17. However, notice that now $j-i = \delta+1$ because $f^{(t)}$ requires the interval to be split. ☐

Lemma 19. *If $f^{(t)}(i,j)$ satisfies concave QI for $j-i=\delta$, then $f^{(1)}(i,j)$ satisfies concave QI for $j-i=\delta$ as well.*

Proof. Since $f^{(1)}(i,j) = c(i,j) = w(i,j) + f^{(t)}(i,j)$, we have that $f^{(1)}(i,j)$ satisfies concave QI as well because it is a sum of two functions which satisfy concave QI. ☐

Lemma 20. *$f^{(1)}, f^{(2)}, \ldots, f^{(t)}$ all satisfy concave QI.*

Proof. By induction on δ and using Lemma 17, Lemma 18, and Lemma 19 together. $\square$

- Now, our main theorem:

Theorem 21. *c as defined in* (3) *can be computed in $O(n^2 \log t)$ time if w is non-negative and satisfy both QI and TI.*

Proof. Let $q_1, q_2, \ldots, q_m$ be an *additive chain* of t; that is, a sequence of integers such that (1) $q_1 = 1$, (2) $q_m = t$, and (3) for any $a > 1$, $q_a = q_b + q_c$ for some $a, b < i$. It is easy to show that we can find an additive change where $m \leq 2 \log t$.

Let $K^{(q_a)}(i,j) = \max_{i \leq k \leq j}\{k \mid f^{(q_a)}(i,j) = f^{(q_b)}(i,k-1) + f^{(q_c)}(k,j)\}$. Using the reasoning similar to that for Lemma 10, we can show that $K^{(q_a)}(i,j) \leq K^{(q_a)}(i,j+1) \leq K^{(q_a)}(i+1,j+1)$ for all $1 \leq a \leq m$.

Hence, we can compute $f^{(q_1)}, f^{(q_2)}, \ldots, f^{(q_m)}$ with Algorithm 2, which runs in $O(n^2 \log t)$ time. $\square$

Algorithm 2 Solve t-ary Recurrence

for $a = 1$ to m **do**
  for $i = 1$ to n **do**
    $f^{(q_a)}[i,i] = 0$
    $K^{(q_a)}[i,i] = i$
  end for
end for

for $\delta = 1$ to $n-1$ **do**
  COMPUTE-F(δ, m)

  for $i = 1$ to $n - \delta$ **do**
    $j = i + \delta$
    $f^{(q_1)}[i,j] = w(i,j) + f^{(q_m)}[i,j]$
  end for

  for $a = 2$ to $m-1$ **do**
    COMPUTE-F(δ, a)
  end for
end for

Algorithm 3 COMPUTE-F(δ, a)

for $i = 1$ to $n - \delta$ **do**
  Let b, c be integers such that $q_b + q_c = q_a$
  $j = i + \delta$
  $f^{(q_a)}[i,j] = -\infty$
  for $k = K^{(q_a)}[i, j-1]$ to $K^{(q_a)}[i+1, j]$ **do**
    if $f^{(q_a)}[i,k-1] + f^{(q_a)}[k,j] \geq f^{(q_a)}[i,j]$ **then**
      $f^{(q_a)}[i,j] = f^{(q_b)}[i,k-1] + f^{(q_c)}[k,j]$
      $K^{(q_a)}[i,j] = k$
    end if
  end for
end for