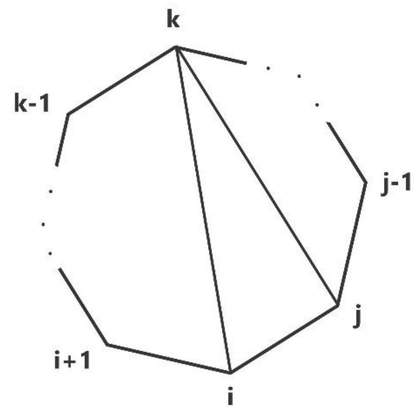
۱

نقاط روی چندضلعی محدب را به صورت ساعت‌گرد از ۱ تا n نام‌گذاری می‌کنیم. حال تابع C را این‌گونه تعریف می‌کنیم که $C(i.j)$ مثلث‌بندی بهینه‌ای می‌باشد که می‌توان با مجموعه $\{i.i+۱. \ldots .j\}$ ساخت. جواب مسئله $C(۱.n)$ می‌باشد.
اگر چند ضلعی محدب i، $i+۱$، ... و j را داشته‌باشیم، در هر حالت ضلع ij در یک مثلث قرار می‌گیرد، چون اگر بخواهد در بیش از یک مثلث قرار بگیرد، در این صورت ضلع‌های آن دو مثلث هم‌دیگر را قطع می‌کنند با توجه به این‌که شکل ما محدب می‌باشد. پس راس دیگر آن می‌تواند $j > k > i$ باشد.



حال می‌توانیم ضلع ij را حذف کنیم و در زیرمسئله‌های $C(k.j)$ و $C(i.k)$ به دنبال مثلث‌بندی بهینه بگردیم. بنابراین $C(i.j)$ برابر عبارت زیر می‌شود:

$$C(i.j) = \begin{cases} |ij|. & j = i+۱ \\ |ij| + \min\{C(i.k) + C(k.j) \mid \forall k :\ j > k > i\}. & j > i+۱ \end{cases}$$

هر مرحله $O(n)$ زمان می‌گیرد و باید برای $O(n^۲)$ خانه این تابع این مرحله را اجرا کنیم پس اردر زمانی $O(n^۳)$ زمان می‌گیرد.

۲

می‌توان ابتدا به دنبال بزرگ‌ترین مربع آبی گشت و دقیقاً همین مراحل را برای پیدا کردن بزرگ‌ترین مربع قرمز دنبال کرد و از مقایسه این دو فهمید که بزرگ‌ترین زیر مربع کدام است. پس ابتدا فرض می‌کنیم به دنبال مربع آبی هستیم و رنگ آبی با ۱ و رنگ قرمز با ۰ مشخص شده است.

ماتریس اولیه را M می‌نامیم و سپس یک ماتریس هم اندازه با M در نظر می‌گیریم و آن را S می‌نامیم. $S(i.j)$ نشان‌دهنده طول ضلع بزرگ‌ترین زیر مربعی که i و j درایه پایین و سمت راست آن در ماتریس M است، می‌باشد. حال اگر $M(i.j) = ۱$ باشد، مقدار $(i.j)$ ماتریس S به صورت زیر تعیین می‌شود:

$$S(i.j) = \min\{S(i-۱.j).S(i-۱.j-۱).S(i.j-۱)\} + ۱$$

در صورتی که $M(i.j) = ۰$ برابر صفر باشد، $S(i.j) = ۰$ می‌شود. بزرگ‌ترین درایه S می‌شود طول ضلع بزرگ‌ترین زیر مربع آبی در ماتریس M. برای رنگ قرمز هم مراحل مشابه را تکرار می‌کنیم و بین دو مقدار به‌دست آمده بزرگ‌ترین را انتخاب می‌کنیم.

می‌خواهیم اثبات کنیم $S(i.j)$ که طول ضلع بزرگ‌ترین زیر مربعی که i و j درایه پایین و سمت راست آن در ماتریس M است را به ما می‌دهد.

در صورتی که $S(i.j) = ۰$ باشد، چون $M(i.j) = ۰$ است پس بزرگ‌ترین زیرمربع طول صفر دارد.

در صورتی که $S(i.j) = k$ باشد، روی k استفرا می‌زنیم.

برای $k = ۱$ داریم عبارت $\min\{S(i-۱.j).S(i-۱.j-۱).S(i.j-۱)\}$ را که برابر صفر شده است. که این معنی را می‌دهد که در یکی از همسایه‌های آن رنگ قرمز مشاهده شده است. پس نمی‌توانیم زیر مربعی با طول ۲ داشته باشیم.

گام استقرا به این صورت است که اگر به ازای k، $S(i.j)$ تعداد درست را نتیجه بدهد برای $k+۱$ باید اثبات کنیم. زمانی که $k+۱$ را داریم باید عبارت $\min\{S(i-۱.j).S(i-۱.j-۱).S(i.j-۱)\}$ برابر k شده باشد. که طبق فرض استقرا درست به جواب رسیده است. معنی این عبارت به این صورت است که در هر همسایه $S(i.j)$ یک زیر مربع با طول حداقل k دیده شده است. پس می‌توان از درایه $(i.j)$ حداکثر به طول $k+۱$ مربع داشت و $S(i.j) = k+۱$ می‌شود.

این الگوریتم بخاطر این که یک جدول $m \times n$ را پر می‌کند، در زمان $O(mn)$ انجام می‌پذیرد.

۳

یکی از رأس‌های این درخت را انتخاب کرده و این رأس ریشه را r بنامید و روی آن BFS را اجرا می‌کنیم تا فرزندان هر رأس به‌دست بیایند. حال برای هر رأس x دلخواه تعریف می‌کنیم:

$A(x) :=$ کم‌ترین تعداد چراغ‌های لازم برای روشن کردن خیابان‌های زیر درخت با ریشه x، اگر در x چراغ باشد.

$B(x) :=$ کم‌ترین تعداد چراغ‌های لازم برای روشن کردن خیابان‌های زیر درخت با ریشه x، اگر در x چراغ نباشد.

$A'(x) :=$ تعداد حالاتی که می‌توان خیابان‌های شهر را با $A(x)$ چراغ روشن کرد، اگر در x چراغ باشد.

$B'(x) :=$ تعداد حالاتی که می‌توان خیابان‌های شهر را با $B(x)$ چراغ روشن کرد، اگر در x چراغ نباشد.

زمانی که در در رأس x چراغ نباشد، در فرزندان این رأس حتماً باید چراغ وجود داشته باشد ولی زمانی که در رأس x چراغ باشد، در فرزندان این رأس آزادی در انتخاب چراغ داریم. برای هر فرزند v رأس دلخواه x، عبارات زیر محاسبه می‌شوند:

$$A(x) = ۱ + \sum \min\{A(v).B(v)\}$$

$$B(x) = \sum \mathrm{A(v)}$$

$$A'(x) = \prod \min\{A'(v).B'(v)\}$$

$$B'(x) = \prod A'(v)$$

به‌طور بازگشتی این عبارات را محاسبه می‌کنیم، هر رأس حداکثر دو بار بررسی می‌شود و زمانی که همه رأس ها را بررسی کنیم الگوریتم تمام می‌شود. پس مرتبه زمانی این الگوریتم $O(n)$ می‌باشد.

حال برای جواب نهایی مسأله اگر $B(r) > A(r)$ باشد، آنگاه با $A'(x)$ حالت می‌توانیم خیابان را روشن کنیم. اگر $B(r) < A(r)$ آنگاه با $B'(x)$ حالت می‌توانیم خیابان را روشن کنیم. و درصورتی که این دو باهم برابر باشند به $A'(r) + B'(r)$ می‌توانیم خیابان را روشن کنیم. چون در این حالت برای کمینه بودن تعداد لامپ‌ها فرقی نمی‌کند که در ریشه لامپ باشد یا نباشد.