শাফায়েতের ব্লগ

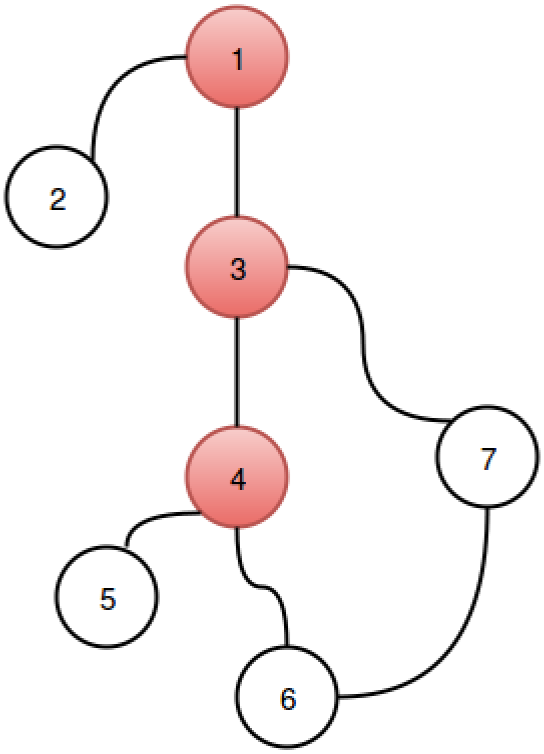
প্রোগ্রামিং, অ্যালগরিদম, ব্যাকএন্ড ইঞ্জিনিয়ারিং

Home
অ্যালগরিদম নিয়ে যত লেখা!
আমার সম্পর্কে...

গ্রাফ থিওরিতে হাতেখড়ি ১৩: আর্টিকুলেশন পয়েন্ট এবং ব্রিজ

নভেম্বর ২৮, ২০১৫ *by Shafaet Ashraf*

আর্টিকুলেশন পয়েন্ট হলো আনডিরেক্টেড গ্রাফের এমন একটা নোড যেটা গ্রাফ থেকে মুছে ফেললে বাকি গ্রাফটুকু একাধিক কম্পোনেন্ট এ ভাগ হয়ে যায়।



উপরের ছবিতে ১, ৩ অথবা ৪ নম্বর নোড এবং সেই নোডের অ্যাডজেসেন্ট এজগুলোকে মুছে দিলে গ্রাফটা একাধিক ভাগ হয়ে যাবে, তাই ১, ৩ ও ৪ হলো এই গ্রাফের আর্টিকুলেশন পয়েন্ট। আর্টিকুলেশন পয়েন্টকে অনেকে কাট-নোড(cut node) , আর্টিকুলেশন নোড বা ক্রিটিকাল পয়েন্ট (critical point) ও বলে।

আর্টিকুলেশন পয়েন্ট বের করার একটা খুব সহজ উপায় হলো, ১টা করে নোড গ্রাফ থেকে মুছে দিয়ে দেখা যে গ্রাফটি একাধিক কম্পোনেন্ট এ বিভক্ত হয়ে গিয়েছে নাকি।

```
procedure articulationPointNaive(G):
     articulation_points=[]
     for all nodes u in G
           G.removeNode(u)
           if get_number_of_component(G)>1
                   articulation_points.add(u)
           end if
           G.addBackNode(u)
      end for
    return articulation_points
```

কম্পোনেন্ট সংখ্যা ডিএফএস বা বিএফএস দিয়ে খুব সহজে বের করা যায়। এই পদ্ধতিতে V বার ডিএফএস চালাতে হবে যেখানে V হলো নোড সংখ্যা, মোট কমপ্লেক্সিটি $O(V \times (V+E))$ বা $O(V^3)$ কারণ সর্বোচ্চ এজ সংখ্যা V^2।

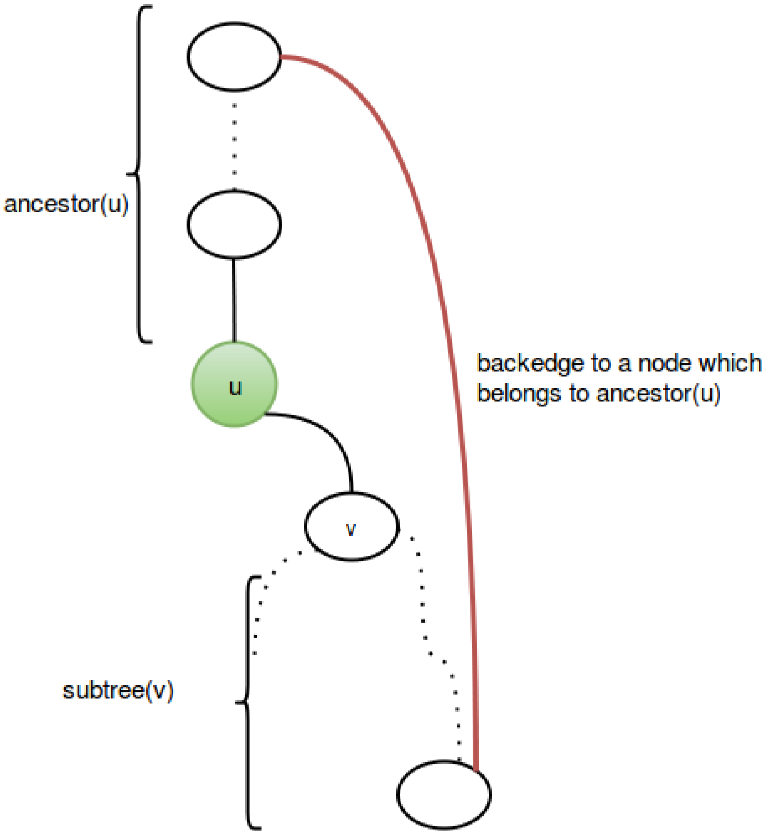
এখন আমরা একবার মাত্র ডিএফএস চালিয়ে আর্টিকুলেশন পয়েন্ট বের করবো। এই অ্যালগোরিদম শেখার জন্য ডিএফএস এর ডিসকভারি/ফিনিশিং টাইম এবং ট্রি এজ ও ব্যাক এজ নিয়ে ধারণা থাকতে হবে।

একটা গ্রাফে ডিএফএস চালালে যেসব ট্রি এজ পাওয়া যায় সেগুলো নিয়ে তৈরি হয়ে ডিএফএস ট্রি।

দুটি কেস থাকতে পারে। যদি একটা নোড ট্রি এর রুট হয় তাহলে একভাবে কাজ করবো, রুট না হলে আরেকভাবে কাজ করবো।

একটা নোড u যদি ট্রি এর রুট হয় এবং ডিএফএস ট্রি তে নোডটার একাধিক চাইল্ড নোড থাকে তাহলে নোডটা আর্টিকুলেশন পয়েন্ট।

রুট ছাড়া বাকি নোডের জন্য কাজটা একটু জটিল।

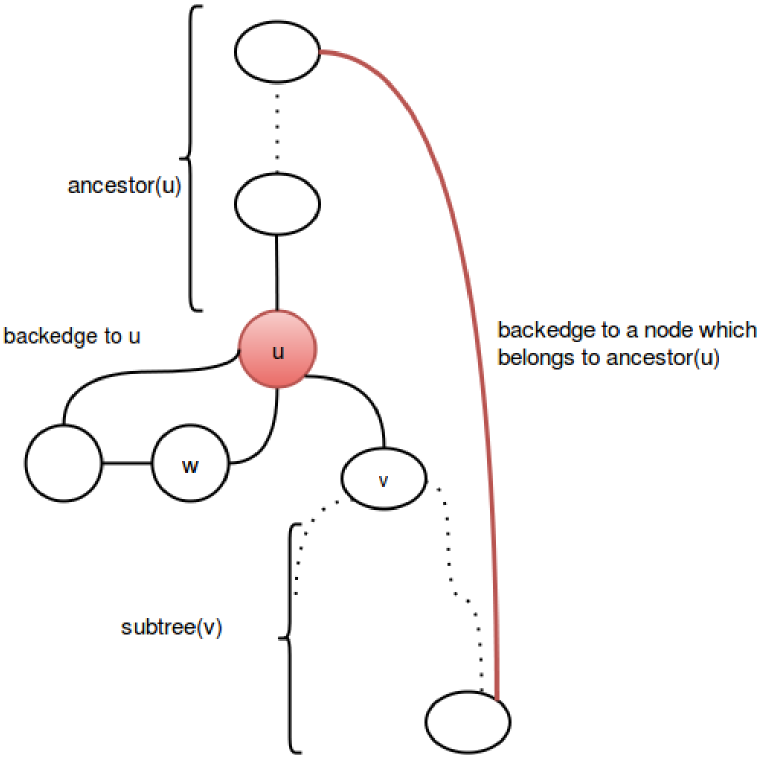


ডিএফএস ট্রি এর একটা এজ $u - v$ এর কথা চিন্তা করো। রুট থেকে u তে আসার পথে যেসব নোড ভিজিট করেছো তাদের আমরা বলবো $ancestor(u)$। এখন v যে সাবট্রি এর রুট সেই সাবট্রির সবগুলো নোডের সেটকে আমরা বলবো $subtree(v)$।

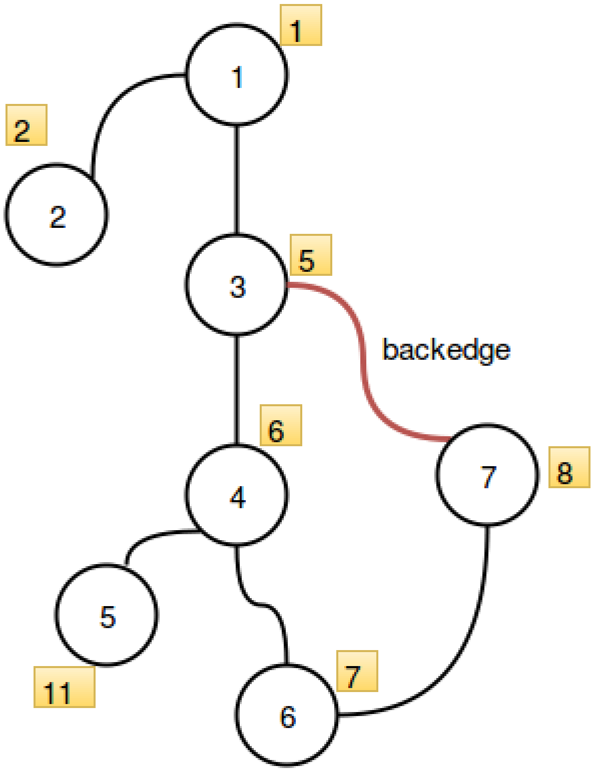
এখন u একটা আর্টিকুলেশন পয়েন্ট হবে যদি মূল গ্রাফে u কে মুছে দিলে $subtree(v)$ এর নোডগুলো একটা আলাদা কম্পোনেন্ট এ পরিণত হয়। $subtree(v)$ আলাদা কম্পোনেন্ট এ পরিণত হবে যদি না মূল গ্রাফে সাবট্রি $subtree(v)$ এর কোনো নোড থেকে ancestor(u) তে একটা ব্যাকএজ থাকে। যদি ব্যাকএজ থাকে তাহলে নোড u এবং অ্যাডজেসেন্ট এজগুলো মুছে গেলেও $ancestor(u)$ থেকে ব্যাকএজ দিয়ে $subtree(v)$ তে পৌছানো যাচ্ছে, নতুন কম্পোনেন্ট তৈরি হচ্ছে না।

u এর যেকোনো একটা চাইল্ড নোড v এর জন্য যদি $subtree(v)$ থেকে $ancestor(u)$ তে পৌছানো না যায় , তাহলে u আর্টিকুলেশন পয়েন্ট, u কে মুছে দিলে সেইসব $subtree(v)$ নতুন কম্পোনেন্ট এ পরিণত হবে যাদের সাথে $ancestor(u)$ এর কোনো ব্যাকএজ সংযোগ নেই।

নিচের ছবিতে $subtree(v)$ যদিও ব্যাকএজ দিয়ে $ancestor(u)$ এর সাথে সংযুক্ত, $subtree(w)$ থেকে $ancestor(u)$ তে ব্যাকএজ নেই। তাই u একটা আর্টিকুলেশন পয়েন্ট।



এবার প্রথম গ্রাফটায় ফিরে আসি। গ্রাফের নোডগুলো ১,২,৩,৪,৬,৭,৫ এই অর্ডারে ভিজিট করলে আমরা প্রতিটা নোডের যা ডিসকভারি টাইম পাবো সেটা পাশে ছোটো করে লেখা হয়েছে:



ডিসকভারি টাইম কিভাবে বের করতে হয় না বুঝলে ডিএফএস নিয়ে টিউটোরিয়ালটা দেখো। d[] দিয়ে আমরা ডিসকভারি টাইম বুঝাবো।

গ্রাফের ব্যাকএজ টা লাল এজ দিয়ে দেখানো হয়েছে। বাকি কালো এজগুলো ডিএফএস ট্রি এর অংশ। 1 হলো রুট নোড।

ডিএফএস ট্রি তে রুট নোড 1 এর চাইল্ড সংখ্যা এখানে ২টা (২ এবং ৩)। তাই 1 একটা আর্টিকুলেশন পয়েন্ট।

লক্ষ্য করো নোড $ancestor(u)$ এর যেকোনো নোডের ডিসকভারি টাইম $d[u]$ এর থেকে ছোটো। আবার u এর অ্যাডজেসেন্ট যেকোনো এজ $u - v$ এর জন্য $subtree(v)$ এর সব নোডের ডিসকভারি টাইম $d[u]$ এর থেকে বড়। এখন $subtree(v)$ এর কোনো নোড থেকে যদি এমন একটা ব্যাকএজ $v - w$ থাকে যেন $d[w] < d[u]$ হয় তাহলে বুঝতে হবে তুমি $u - v$ এজ পার হয়ে $subtree(v)$ দিয়ে $ancestor(u)$ তে পৌছে গেছো এবং $w \in ancestor(u)$ । তারমানে u মুছে দিলেও $subtree(v)$ থেকে w তে পৌছানো যাবে।

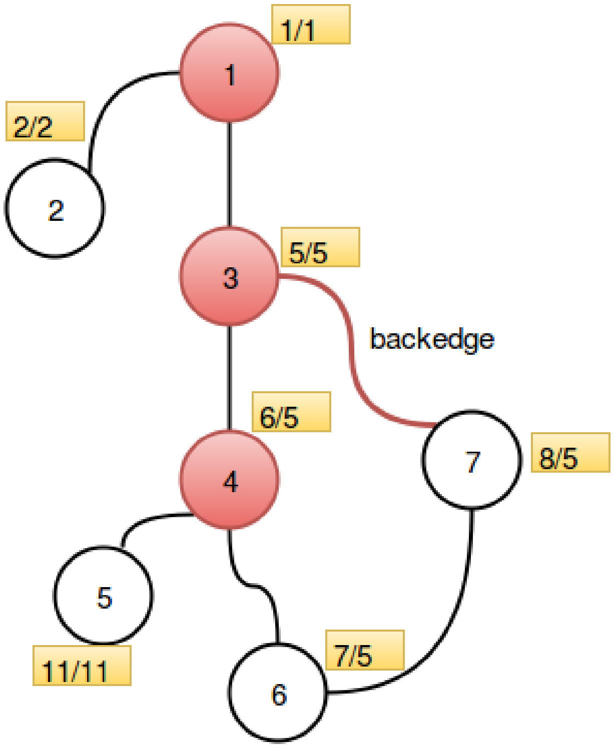
যেমন 4 নম্বর নোডের কথা চিন্তা করো। 4 এর ডিসকভারি টাইম $d[4] = 6$ এবং $ancestor(4) = \{1, 2, 3\}$। এখন 4-6 এজটার কথা ভাবি। $subtree(6)$ এ একটা ব্যাকএজ $7 - 3$ আছে, এবং $d[3] = 5$ যা $d[4]$ এর থেকে ছোটো। তারমানে $3 \in ancestor(4)$। তাহলে তুমি 4 নোডটা মুছে দিলেও $subtree(6)$ ব্যাকএজের মাধ্যমে $ancestor(4)$ এর সাথে সংযুক্ত থাকবে।

এবার আমরা আরেকটা ভ্যারিয়েবল ডিফাইন করবো $low[u]$। মনে করো $subtree(u)$ এবং $subtree(u)$ এর সাথে ব্যাকএজ দিয়ে সংযুক্ত সবগুলো নোডের একটা সেট বানানো হলো, সেটা টা হলো $\{x1, x2 \ldots xm\}$। তাহলে $low[u]$ হবে $min(d[x_1], d[x_2] \ldots, d[x_m])$।

যেমন 4 নম্বর নোডের জন্য $subtree(u) = \{5, 6, 7\}$ এবং $subtree(u)$ এর সাথে ব্যাকএজ দিয়ে যুক্ত আছে নোড 3। তাহলে $low[u] = min(d[5], d[6], d[7], d[3]) = 5$।

এখন চিন্তা করো কোনো একটা এজ $u - v$ এর জন্য d[u]>low[v] হবার অর্থ কি? d[u] এর থেকে ডিসকভারি টাইম ছোটো একমাত্র $ancestor(u)$ সেটের নোডগুলোর। subtree(v) এর কোনো নোড ব্যাকএজ দিয়ে ancestor(u) এর সাথে যুক্ত, সেজন্য low[v] এর মান d[u] এর থেকে কমে গিয়েছে। যদি $d[u] <= low[v]$ হয়, তাহলেই শুধুমাত্র u একটা আর্টিকুলেশন পয়েন্ট হবে।

আগের গ্রাফেই ডিসকভারি টাইমের পাশাপাশি $low[u]$ এর মানগুলোও দেখি:

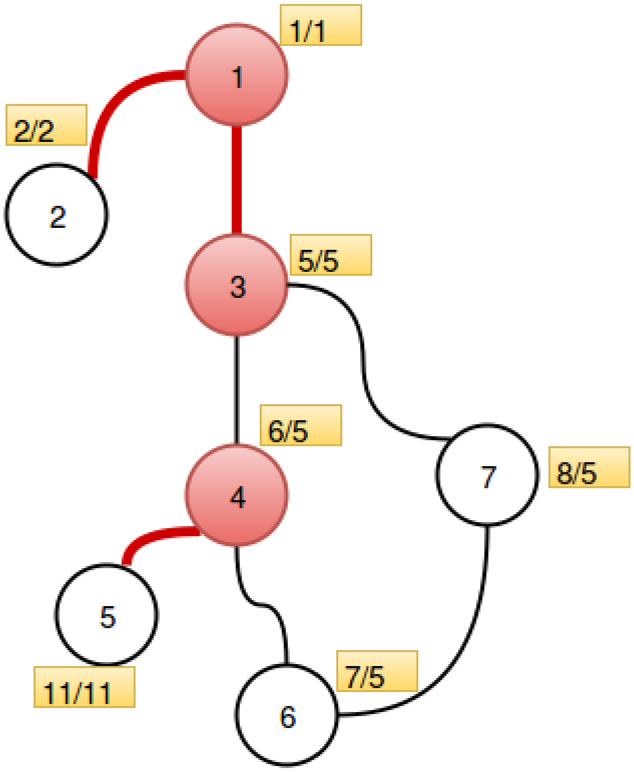


তাহলে আমরা আর্টিকুলেশন পয়েন্ট বের করার একটা অ্যালগোরিদম পেয়ে গিয়েছি। প্রতিটা নোডের জন্য d[u], low[u] বের করতে পারলেই কাজ শেষ। low[u] বের করা কঠিন কিছু না, সুডোকোড দেখলেই পরিষ্কার হবে:

```
articulation_point[] ← false
visited[] ← false
low[]=d[u] ← 0
time ← 0
Procedure FindArticulationPoint(G, u):
time ←  time+1
low[u]=d[u] ← time
visited[u] ← true
no_of_children ←  0
for each edge u to v in G.adjacentEdges(u) do
   if(v == parent[u]) continue
   if visited[v] //This is a backedge
      low[u] = min(low[u], d[v])
   end if
   if not visited[v]   //This is a tree edge
      parent[u] = v
      FindArticulationPoint(G, v)
```

```
14          low[u] = min(low[u], low[v])
15          if d[u] <= low[v] and u is not root:
16             articulation_point[u]=true
17          end if
18          no_of_children=no_of_children+1
19       end if
20       if(no_of_children>1 u is root):
21            articulation_point[u]=true
22       end if
23    end for
```

ব্রিজ জিনিসটা আর্টিকুলেশন পয়েন্টের মতই। গ্রাফ থেকে যে এজ তুলে দিলে গ্রাফটা একাধিক কম্পোনেন্টে ভাগ হয়ে যায় তাকেই বলা হয় ব্রিজ।



উপরের গ্রাফে $4-5$, $1-2$, আর $1-3$ এই ৩টি এজ হলো ব্রিজ।

ব্রিজ আর আর্টিকুলেশন পয়েন্টের সুডোকোডের পার্থক্য খালি এক জায়গায় ১৫ নম্বর লাইনে d[u]<=low[v] এর জায়গায় d[u]<low[v] লিখতে হবে। এটা কেন কাজ করে তুমি সহজেই বুঝতে পারবে যদি তুমি সুডোকোডটা বুঝে থাকো, তাই আর ব্যাখ্যা করলাম না।

দুটি নোডের মধ্যে একাধিক এজ থাকলে অবশ্য এটা কাজ করবে না। তখন কি করতে হবে সেটা চিন্তা করা তোমার কাজ!

সলভ করার জন্য কিছু প্রবলেম পাবে এখানে।

হ্যাপি কোডিং!

ফেসবুকে মন্তব্য