برنامه ریزی کارها برای کارگران غیر قابل اعتماد

توسط

سارا احمدیان

رساله ارائه شده به عنوان بخشي از ملزومات براي دريافت درجه كارشناسي مهندسي نرم افزار

زير نظر

جناب آقاي دكتر قدسي

شهريور ۸۷

دانشكده ي مهندسي كامپيوتر

دانشگاه صنعتي شريف

تهران

تقدیم به پدر ومادر عزیزم

و

دوستانم مرضیه وسیاوش.

قدرداني

در اين جا بر خود لازم مي دانيم از تمامي افرادي كه در انجام اين پروژه ما را ياري نمودند، به خصوص از استاد ارجمند، جناب آقاي دكتر قدسي، و آقاي مصطفى نورى كه در تمام مراحل انجام اين پروژه با راهنمايي هاي بي دريغ خود ما را ياري كردند، تشكر نماييم.

چکیده

در چندسال اخیر محاسبات شبکه اي[1] براي تسريع انجام محاسبات بر روي يک مجموعه از کامپيوتر ها گسترش يافته است. يکي از بزرگترين دغدغه[2] هاي موجود در اين زمينه دستيابي به کارايي با اجراي محاسبات در مقياس بزرگ[3] و پيچيده[4] بر روي ماشين هاي غير قابل اعتماد[5] است.در اين مورد مساله هاي مختلفي مطرح مي شود . براي نمونه هنگامي که يک کامپيوتر در انجام يک کار شکست مي خورد، فر آيند انجام کارها به تاخير مي افتد زيرا کارهاي وابسته به اين کار تا اتمام موفق اين کار، نمي توانندشروع شوند.

مساله مشابهي درمديريت پروژه[6] مانند توليد يک محصول يا توسعه ي يک نرم افزار مطرح مي شود. در اين جامجموعه از کار ها و دسته از پيش نيازي ها داريم. کارگر هامي توانند به انجام کار ها منسوب شوند. در عمل ممکن است کارگري که براي انجام کاري تعيين مي شود، شکست بخورد. براي مثال اگر فعاليت مورد نظر از دو بخش نوشتن کد و تست آن تشکيل شده باشد ممکن است در قسمت تست شکست بخورد. مدير پروژه بايد توانايي تقريب احتمال موفقيت کارگران را داشته باشد.

[1] Grid computing

[2] Challenge

[3] Large-scale

[4] Sophisticated

[5] unreliable

[6] Project management

با توجه به کاربرد عملي مساله "برنامه ريزي چند پردازه اي"[1] باسناريوهاي در محاسبات شبکه اي و مديريت پروژه، بر آن شديم که آن را برررسي کنيم. مساله برنامه ريزي چند پردازه اي در عدم قطعيت، اولين بار توسط مالويکز [1] معرفي شد. در مساله ي برنامه ريزي چند پردازه اي، در انجام صحيح کارهاي موکول شده به پردازنده ها عدم قطعيت[2] وجود دارد. نام ديگر اين مساله برنامه ريزي کارگران غير قابل اعتماد است. مساله برنامه ريزي کارها براي کارگران غير قابل اعتماد، يک مساله برنامه ريزي موازي[3] است.

مساله اول اين پايان نامه، مساله برنامه ريزي کارگران غير قابل اعتماد است. در اين مساله، t کار تک واحد زماني و n کارگر داريم. وابستگي بين کارها با يک "گراف جهت دار بدون دور"[4](DAG) مدل شده است به علاوه هر کارگر i کار j را با احتمال $P_{i,j}$ به درستي انجام مي دهد. هدف يافتن انتساب بهينه ي[5] $\sum$ است که نشان دهد در هر مرحله کارگران چگونه به کارها منسوب شوند (احتمالا به‌صورت موازي و تکراري) تا زمان اجراي کارها کمينه شود.

مالويکز [1] نشان داده است که براي وجود راه حل چند جمله اي براي مساله بالا بايد تعداد کارگران و عرض گراف[6] کارها عدد ثابت باشد. اين دو محدوديت شايد بسيار شديد به نظر برسند، اما احتياج هستند. مالويکز [1] اثبات مي کند در حالتي که حتي يکي از اين دو ثابت نيست، مساله سخت-تمام[7] است.

---

[1] Multiprocessor scheduling

[2] Uncertainty

[3] Parallel scheduling

[4] Directed Acyclic Graph(DAG)

[5] Optimal assignment

[6] Graph width

[7] NP-hard

مساله دوم معرفي شده، برنامه ريزي كارگران ماركوفي ، حالت كليتري از مساله اول است. در اين مساله علاوه بر ورودي هاي مساله اول، عملكرد هر كارگر در هر زمان وابسته به "زنجيره ي ماركوف زمان گسسته"[1] آن كارگر است. در اين حالت احتمال انجام كار i توسط كارگر j در طول زمان ثابت نيست و وابسته به $P_{i,j}$ و زنجيره ي ماركوف آن است. هدف يافتن برنامه ريزي بهينه ي $\sum$ است كه نشان دهد در هر مرحله كارگران چگونه به كارها منسوب شوند تا زمان اجراي كارها كمينه شود.

براي وجود راه حل چند جمله اي براي مساله دوم، علاوه بر تعداد كارگران و عرض گراف كارها،طول دورها در گراف القاشده[2] بر روي يال هاي شكست نيز عدد ثابت باشد. در نهايت پياده سازي از الگوريتم هاي چند جمله اي براي مساله ها باشرايط محدود ارايه شده است.

---

[1] Discrete time markov chain

[2] Induced graph

فهرست مطالب

فهرست شکل ها

۱مقدمه

مساله برنامه ریزي، انتساب مجموعه از کارها به کارگران مختلف درطول زمان است. با ترکيب شرايط مختلف کارگران، هدف هاي مختلف و شرايط محدود کننده ي ديگر مي توان مساله هاي مختلفي در اين مورد مطرح کرد. در نتيجه ي اين محدوديت هاي مختلف و با توجه به کارهاي مداوم پژوهشگران در اين زمينه در دهه هاي اخير، ادبيات برنامه ريزي کم کم غني شده است و درطول زمان به يک رشته علمي بالغ تکامل يافته است.

مساله هاي برنامه ريزي گوناگوني در گذشته مورد مطالعه قرار گرفته اند که از رده ي مساله هاي ساده [1] تا مساله هاي سخت-تمام [3] هستند. در گذشته، هدف در اين مساله هاي کمينه کردن زمان تکميل يا بيشينه کردن سود است. در يک حيطه جديد، احتمال طبيعي شکست نيز در نظر گرفته مي شود. در اين حيطه کارگران غير قابل اعتماد هستند يعني مي توانند در انجام يک کار با يک احتمالي

شکست بخورند. پيش نيازي کارها نيز در برنامه ريزي در نظر گرفته مي‌شود که اين پيش نيازي ها مي تواند از يک دنباله ي ساده تا يک گراف جهت دار بدون دور عمومي رده بندي شوند[1,4].

۱-۱ کارهاي پيشين

مطالعات مختلفي در زمينه برنامه ريزي با منابع محدود صورت گرفته است نارسيمهان[5] ، يک مرور کامل در اين مورد کرده است. اين مساله هاي معمولا سخت-تمام هستند و مطالعات بيشتر در جهت يافتن روش حريصانه[1] هستند. مساله هاي مطرح در اين پايان نامه از جهت در نظر گرفتن احتمال شکست کمي با اين مساله هاي متفاوتند. از بين کارهاي انجام شده کار انجام شده توسط موري [6] با توجه به اين که فرض کرده است هر کار ممکن است به روش هاي مختلفي انجام شود به اين کارها نزديکتر است. هر روش احتياج به منابع مختلف ،طول زمان خاص و احتمال شکست متفاوتي دارد. اگر کار با شکست مواجه شود در نوبت بعدي حتما به درستي انجام مي شود. هدف کمينه کردن زمان تکميل است. براي اين مساله راه حل هاي حريصانه اي پيشنهاد شده است. مدل مشابهي توسط تسنگ [7] با هدف بيشينه کردن احتمال اتمام موفق به صورت حريصانه داده است.

در برنامه ريزي پروژه در عدم قطعيت[8,9,10] هر کار يک طول زمان تصادفي براي انجام و احتياج به منابع مختلفي براي انجام دارد. تعداد ثابتي از هر منبع موجود است. هدف يافتن انتساب بهينه ي منابع به کارها براي داشتن زمان اتمام کمينه است. مساله هاي ما شبيه اين مساله هاي هستند

[1] Heuristic approach

چون کارگر ها رامي توان به عنوان منبع در نظر گرفت. اما ازطرفي هم با اين مساله متفاوتند زيراما اجازه مي دهيم کارگران مختلف به يک کار منسوب شوند که توزيع زمان انجام کارها را تغيير مي دهد.تورنکويست [11] برنامه ريزي کارها باطول زمان متفاوت که بسته به نحوه ي توزيع منابع دارد رامعرفي کرده است. انجام کار باشکست مواجه مي‌شود اگرطول زمان انجام کار از آستانه[1] بيشتر شود.هدف بيشينه کردن احتمال اتمام است.

يکي از اهداف برنامه ريزي تصادفي[2] کمينه کردن زمان تکميل مورد انتظار[3] در حالتي که‌طول زمان اجراي کارها متغير و کارها بر روي ماشين هاي موازي برنامه ريزي مي شوند. اين مساله هاي توسط کلينبرگ [14]و گوبل [15] براي کارهاي مستقل[4] و اسکوتلا [16] براي کارهاي وابسته مطالعه‌شده اند. اما اين‌مساله هاي از اين نظر که‌ما فرض‌مي کنيم‌منابع مختلف‌مي توانند به يک کار تخصيص داده‌شوند بامساله هاي‌مامتفاوتند.

---

[1] Threshold

[2] Stochastic

[3] Expected completion time

[4] Independent

۱-۳ساختار پایان نامه

در فصل دوم مساله اول یعني برنامه ریزي کارگران غیر قابل اعتماد بررسي شده است. در این فصل ابتدا یک سري تعریفات پایه ،سپس توضیح دقیق مساله آنگاه توضیح الگوریتم آن وسپس تحلیل درستي آن و در نهایت تحلیل زمان اجراي آن آمده است.

در فصل سوم مساله دوم یعني برنامه ریزي کارگران غیر قابل اعتماد با زنجیره ي مارکوفي یا به صورت ساده تر یعني برنامه ریزي کارگران مارکوفي بررسي شده است. در این فصل ابتدا یک سري تعریفات پایه ،سپس توضیح دقیق مساله آنگاه توضیح الگوریتم آن و در نهایت تحلیل زمان اجراي آن آمده است.

در فصل ۴ اثبات سخت-تمام بودن مساله هاي آورده شده است. فصل ۵ نیز در رابطه با توضیح پیاده سازي الگوریتم ها و تحلیل زمان اجراي آنها است.

۲مساله برنامه ریزي کارگران غیر قابل اعتماد

۲-۱ تعریف هاي پایه

گراف جهتدار G یك مجموعه از رلس ها N_G و کمان[1] ها (یال هاي جهتدار) A_G به فرم $(u \rightarrow v)$ که $u,v \in N_G$، مي باشد یك مسیر[2] در G، دنباله اي از کمان ها است که یالهاي پشت سر هم، انتهاي یکي ابتداي دیگري است و هیچ رلس تکراري درمسیر نداریم مگر در ابتدا و انتهاي آن. براي مثال مسیر از رلس u_1 به u_k (درصورتي که $u_k=u_1$ در این صورت مسیر، دور[3] است):

$$(u_1 \rightarrow u_2)(u_2 \rightarrow u_3)....(u_{k-2} \rightarrow u_{k-1})(u_{k-1} \rightarrow u_k)$$

"گراف جهتدار بدون دور"(DAG) ، G، گراف جهتداري است که در آن دور نداریم. براي کمان $u \rightarrow v \in A_G$، u پدر[4] v و v فرزند[5] u است. هر رلس بدون پدر در G، چشمه[1] و هر رلس بدون فرزند چاه[2] است. وسایر رلس ها، رلس هاي داخلي[3] هستند.

---

[1] Arc

[2] Path

[3] Cycle

[4] Parent

[5] Child

در يك گراف جهتدار بدون دور، يك پاد زنجير[4] مجموعه اي از رئوس هاست كه دوبه دو مقايسه-پذير[5] نيستند يعني براي دو رأس مجزاي u,v در اين پادزنجير هيچ مسيري از u به v يا v به u وجود ندارد. عرض گراف[6] برابر اندازه بزرگترين پادزنجير است.

يك زنجير[7] يك مسير است. يك مجموعه از زنجيرها گراف را پوشش مي دهد درصورتي كه هر رأس از گراف حداقل در يك زنجير آمده باشد (زنجيرها ممكن است اشتراك[8] داشته باشند.)

قضيه معروف ديلورث [3] بيان مي دارد كه تعداد زنجيره لازم براي پوشاندن گراف برابر عرض گراف است.

---

[1] Source

[2] Sink

[3] Internal node

[4] Antichain

[5] Comparable

[6] Graph width

[7] Chain

[8] Overlap

مساله با يك گراف جهتدار بدون دور مدل شده است كه رئوس ها در آن كارها هستند $(N_G = \{1,2,...,t\})$ ومجموعه $\{1,2,...t\}$ را با [t] نمايش مي دهيم. كمان ها نشان دهنده وابستگي بين كارها هستند. براي مثال $u \to v$ بيانگر اين است كه v نمي تواند اجراشود مگر آنكه u درمراحل قبلي به‌صورت موفقيت آميز انجام‌شده باشد.

مجموعه اي از رئوس ها شرط پيش نيازي[1] را دارند اگر براي هر كار t درمجموعه تمام پدران اين كار نيز درمجموعه باشند. اگر X مجموعه اي باشد كه اين‌شرط را داشته باشد، E(x) مجموعه از كارها است كه در مجموعه نيستند اما همه پدران آنها در مجموعه قرار دارند. هنگامي كه كارهاي مجموعه X اجراشوند مجموعه E(x) مجاز[2] خوانده مي‌شود (بنابراين هر چشمه اي كه در X نباشد مجاز خوانده‌مي‌شود).

اجراي كارها به‌صورت يك بازي‌مدل مي‌شود. n كارگر كه با مجموعه [n] مشخص مي‌شوند داريم. X مجموعه اي است كه‌شرط پيش نيازي را دارد و درشروع Y=X قرارمي دهيم و بازي در دورها ادامه مي يابد. در هر دور كارگرها براي كارهاي موجود در E(y) با توجه به برنامه ريزي Σ معين مي‌شوند. در واقع برنامه ريزي هر كارگر را به يك كار از مجموعه $E(y) \cup \perp$ نسبت مي دهد اگر كارگر به $\perp$ منسوب شود يعني در اين دور كارگر استراحت مي كند. توجه كنيد كه انتساب

[1] Precedence constraint

[2] Eligible

کارگرها با توجه به Y تعيين مي شود. در انتساب کارگران ممکن است چند کارگر به يك کار منسوب شوند. سپس هر کارگر سعي مي کند کاري را که به آن منسوب شده است انجام دهد. تلاش کارگر i که به کار j منسوب شده با احتمال $0 \le P_{i,j} \le 1$ به‌صورت مستقل از ساير تلاش ها موفقيت آميز خواهد بود. يك کار در يك دور انجام مي‌شود اگر حداقل يك کارگر منسوب شده به آن کار آن را درست انجام دهد. تمام کارهاي انجام‌شده در اين دور به Y اضافه مي‌شوند و بازي در دورها ادامه مي يابد. ممکن است در يك دور تمام تلاش ها با شکست مواجه شوند در اين‌صورت Y بدون تغيير باقي مي‌ماند و در نتيجه در دور بعدي انتساب کارگرها بدون تغيير خواهد بود.

به‌صورت رسمي، يك انتساب يك تابع به‌صورت:

$$\sum : 2^{N_G} \rightarrow [n] \rightarrow (N_G \cup \{\perp\})$$

به‌طوري که براي هر زيرمجموعه Z که شرط پيش نيازي را دارا باشد Σ يك انتساب از [n] به $E(z) \cup \{\perp\}$ است. بازي آنقدر ادامه مي يابد تا تمامي چاه ها در Y قرار بگيرند. در اين دور ما مي گوييم که بازي تمام‌شده است.

تعداد دورهايي که لازم است تا بازي تمام‌شود، زمان تکميل انتساب Σ باشروع از مجموعه X مي نامند. اين زمان يك متغير تصادفي است. زماني که X تهي باشد آن را به‌صورت ساده زمان تکميل مي نامند. هدف پيدا کردن انتساب *Σ که زمان تکميل مورد انتظار را کمينه مي کند.

۲-۲ تعریف مساله :

مساله یك گراف كه بیانگر وابستگي كارها و n كارگر به طوري كه كارگر i با احتمال $0 \le P_{i,j} \le 1$ كار j را به درستي انجام مي دهد و براي هر كار j حداقل یك كارگر وجود دارد كه $0 < P_{i,j}$.

هدف پیدا كردن برنامه ریزي بهینه Σ^* است كه «زمان تكمیل» را كمینه كند.

فرض كنیم n زیرمجموعه اي از كارها است كه شرط پیش نیازي را دارد. B_X نشان دهنده زمان اجراي كمینه براي تكمیل كارها با شروع از X است. مالوبكز [1] نشان مي دهد این زمان متناهي است.

***لم ۱:** براي مجموعه X از كارها كه شرط پیش نیازي دارد، B_X متناهي است.*

اثبات: انتسابي معرفي مي‌شود كه زمان تكمیل آن متناهي است و بنابراین زمان تكمیل كمینه نیز متناهي است.

فرض كنیم دنباله ترتیب مكاني[1] از كارهاي انجام نشده به صورت $t_1,t_2,...,t_m$ باشد. توجه كنید كه مي توان كارها به ترتیب این دنباله اجرا شوند چرا كه وقتي كارهاي $t_1,t_2,...t_{j-1}$ اجرا مي‌شوند

[1] Topological sort

کار t_i، مجاز است. انتساب این ترتیب را دنبال مي کند و در هر مرحله همه کارگرها را به اولین کار انجام نشده از این دنباله منسوب مي کند. احتمال اینکه تمام کارگرها در انجام کار t_j با شکست مواجه شوند برابر $\prod_{i=1}^{n}(1-P_{i,t_j})$ است که با توجه به فرض حتماً از یك اکیدا کوچکتر است. بنابراین زمان مورد انتظار براي اجراي این کار با توجه به اینکه احتمالات توزیع هندسي دارند برابر $\frac{1}{(1-\prod_{i=1}^{n}(1-P_{i,t_j}))}$ است. با استفاده از خطي بدون زمان اجراي مورد انتظار مي توان نتیجه گرفت زمان اجراي مورد انتظار کل جمع این زمان اجراهاست و بنابراین حداکثر برابر $\sum_{j=1}^{t}\frac{1}{(1-\prod_{i=1}^{n}(1-P_{i,t_j}))}$ است.

¤

۳-۲ الگوریتم برنامه ریزي براي مدل محدود مساله:

مالویکز [1] یك الگوریتم با زمان اجراي چند جمله اي براي پیدا کردن انتساب بهینه Σ^* در مدل محدود مساله ارائه داده اند. بنابراین دو فرض داریم

1. عرض گراف جهت دار G یك عدد ثابت است
2. تعداد کارگران عدد ثابت است.

شاید این دو محدودیت بسیار شدید به نظر برسند ولي اثباتي در ادامه ارائه شده که نشان مي دهد که هر دوي این شرط ها لازم است.

در ابتدا براي زمان مورد انتظار يك رابطه بازگشتي ارائه شده است. اين رابطه وابستگي يك زمان براي مجموعه داراي شرط پيش نيازي را با مجموعه هاي بزرگتر نشان مي دهد و اين رابطه پايه اي براي برنامه داينامیک است.

قضيه ۳: *انتساب Σ، مجموعه X كه شرط پيش نيازي را دارد و تمامي چاه هاي G در آن قرار ندارد را د نظر بگيريد. فرض كنيد زمان تكميل مورد انتظار براي براي انتساب Σ با شروع از X متناهي باشد. مجموعه هاي $D_0, D_1, \ldots, D_k$ تمامي زير مجموعه هاي مجزاي[1] E(X) هستند به طوري كه $D_0 = \phi$.*

احتمال اينكه D_i دقيقاً مجموعه اي از كارها باشد كه در انتساب Σ توسط كارگران به درستي انجام مي شوند برابر a_i است و $X_i = X \cup D_i$. آنگاه:

$$T_{X_0} = \frac{1}{a_1 + a_2 + ... + a_K}(1 + \sum_{i=1}^{k} a_i T_{X_i})$$

كه T_{X_i} زمان تكميل مورد انتظار انتساب Σ با شروع از X_i است.

اثبات: متغير تصادفي T را زمان تكميل انتساب Σ با شروع X در نظر مي گريم. اين متغير برابر زمان مورد انتظار اجراي كاري خارج از X بعلاوه زمان تكميل مورد نياز براي اتمام ساير كارهاست.

[1] Distinct subsets

قسمت اول داراي توزيع هندسي دارد و قسمت دوم مي تواند به صورت ترکب خطي T_{X_i} محاسبه شود. چرا که وقتي که کارهاي D_i انجام مي‌شود زمان تکميل انتساب Σ براي X_i برابر T_{X_i} است.

پس براي تعيين زمان تکميل مورد انتظار براي اينکه x شامل تمامي چاه هاي G شود. سناريوي زير تکرار مي‌شود.

۱- کارگران با انتساب $\sum(X)$ به کارهاي E(x) منسوب شوند.

۲- کارگران تلاش کنند تا کارهاي خود را انجام دهند.

۳- کارهاي انجام شده در مرحله ۲ به X اضافه شوند.

۴- هدف پيدا کردن زمان تکميل مورد انتظار است.

با توجه به سناريوي بالا پس از يک واحد زماني به احتمال a_i در حالت X_i هستيم پس مي توان گفت:

$$T_{X_0} = 1 + \sum_{i=0}^{k} a_i T_{X_i}$$

در نتيجه:

$$(1 - a_0) * T_{X_0} = 1 + \sum_{i=1}^{k} a_i T_{X_i}$$

بايد نشان دهيم $(1-a_0)$ برابر صفر نيست. $(1-a_0)=a_1+a_2+...+a_k$ برابر است با احتمال اينكه حداقل يك كار از $E(x)$ اجرا شود. (توجه كنيد $E(x)$ غيرتهي است) و از آنجايي كه با توجه به فرض T_x متناهي است، پس $(1-a_0)$ مخالف صفر است. در بازنويسي معادله بالا داريم:

$$T_{X_0}=\frac{1}{a_1+a_2+...+a_k}(1+\sum_{i=1}^{k}T_{X_i})$$

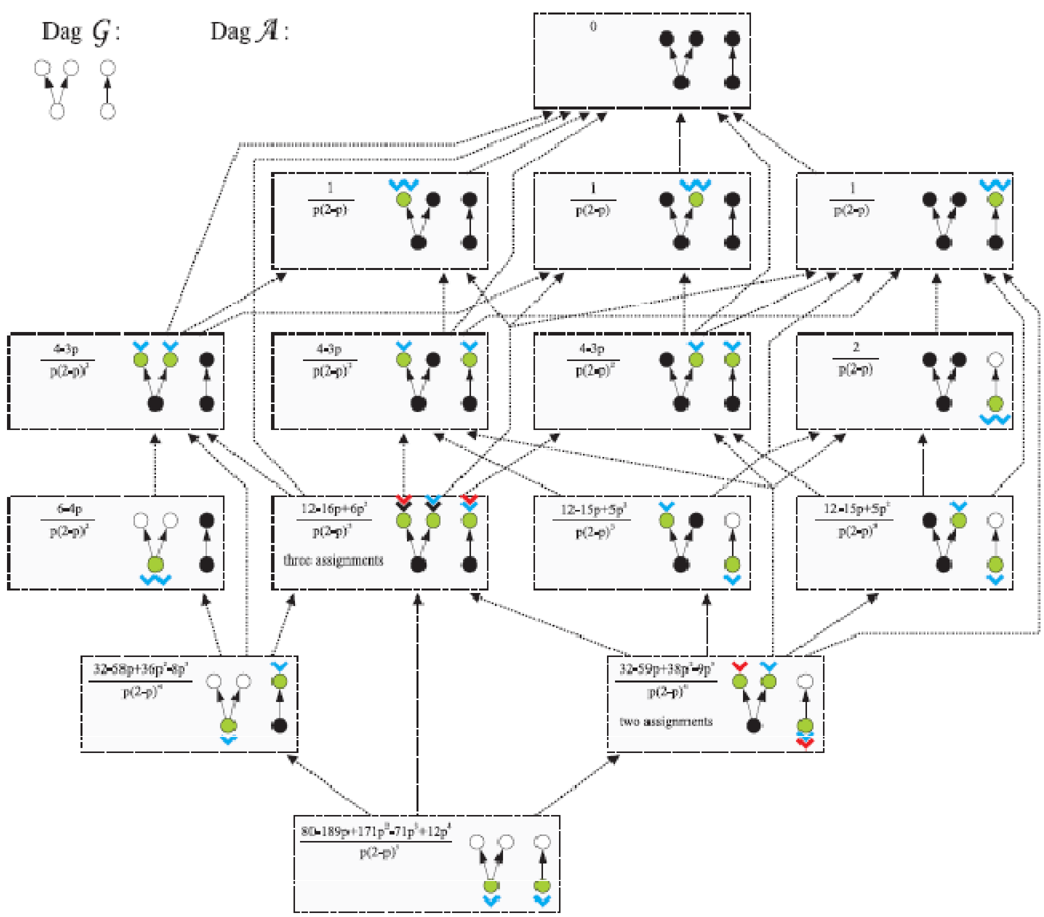
¤

به نظر مي رسد وابستگي كه بين Xهاي مختلف وجود دارد به شكل يك گراف جهت دار بدون دور است. گراف جهتدار از $A(N_A,A_A)$، گراف تكامل مجاز[1] اجراي G را بدين صورت مي سازيم (مثال در شكل ۱):

گراف A را به صورت استقرايي مي سازيم. هر راس از A نشان دهنده زير مجموعه اي از رئوس G است. كار را با مجموعه $N_A=\{\phi\}$ آغاز مي كنيم. در هر مرحله براي هر راس از $X\in N_A$ كه شامل تمام چاله هاي G نمي باشد، $E(x)$ را محاسبه مي كنيم. حال تمام زير مجموعه هاي غيرتهي $D\in E(x)$ را در نظر مي گيريم و درصورت عدم وجود راس $X\cup D$ آن را به مجموعه N_A اضافه كرده كمان $(X,X\cup D)$ را به مجموعه اضافه مي كنيم.

[1] Admissible evolution

از آنجايي كه G محدود است، فرايند استقرايي به‌صورت واضح يك گراف جهتدار يكتا تعيين مي كند. ساختار اين گراف در لم بعدي تشريح شده است.



شكل ۷-درشكل گراف تكامل مجاز ساخته‌شده از گراف وابستگی کارهای G

لم ۳-۲: *فرض كنيم A گراف تكامل مجاز گراف G باشد آنگاه A گراف جهت دار بدون دور است و مسير رلس آن دقيقاً يك مجموعه از كارهاست كه شرط پيش نيازي را دارد. A دقيقاً يك چشمه و يك چاه دارد.*

اثبات: گراف A بدون دور است زیرا هر کمان در A از رلس X به $X \cup D$ به مجموعه‌ي با سایز اکیداً بزرگتر از X است، مي باشد.

هر رلس از Aشرط پیش نیازي را دارد. در واقع اگر Xشرط پیش نیازي را داشته باشد، اجتماع آن با هر زیرمجموعه از E(X) آن نیز این شرط را دارد. پس هر رلس از Aشرط پیش نیازي را دارد.

حال هر مجموعه Y از کارها که شرط پیش نیازي را دارد در نظر بگیرید. حال کارهاي $t_1,t_2,...,t_{|Y|}$ از Y را که به ترتیب مکاني آمده اند را در نظر بگیرید. به صورت واضح t_j به مجموعه ی مجاز از کارها هنگامي که کارهاي $t_1,t_2,...,t_{j-1}$ انجام شده اند، تعلق دارد. بنابراین اگر $\{t_1,t_2,....,t_{j-1}\}$ یك رلس از A است، $\{t_1,t_2,....,t_j\}$ نیز یك رلس از A است. ϕ یك رلس از A است، بنابراین Y نیز یك رلس از A است. نتیجه دیگر این است که N_G نیز یك رلس از A است.

ما رلس Y را به رلس هاي A اضافه مي کنیم اگر یك کمان به Y وجود داشته باشد. بنابراین Y نمي تواند چشمه باشد. از طرفي ϕ یك چشمه است چرا که هیچ کماني به مجموعه اي با اعضاي برابر یا کمتر نداریم و سایز ϕ، صفر است.

رلس X از رلس هاي A را در نظر بگیرید؛ فرض کنید همه چاه هاي G در X نباشند. با توجه به ترتیب مکاني رلس هاي Gما مي بینیم که رلسي وجود دارد که همه پدرهاي آن در Xقرار دارند و خود آن در X نیست. بنابراین E(X) غیر تهي است و X یك فرزند در A دارد. حال رلس X که شامل تمام چاه ها G است را در نظر بگیرید. از آنجایي که Xشرط پیش نیازي را دارد بنابراین Xشامل تمام رلس هاي G است و بنابراین $X=N_G$. E(x) تهي است بنابراین Xهیچ فرزندي ندارد.

¤

از آنجايي که A هيچ دوري ندارد، مي توانيم قضيه ۲ را با شروع از يك چاه و حركت رو به عقب[1] به چشمه، اعمال كرد. مالويكز [1]، يك الگوريتم داينامیك به نام OPT براي پيدا كردن انتساب بهينه $\sum$ ارائه مي كند. فرض كنيد $Y_1, Y_2, \ldots, Y_m$، ترتيب مكاني رلس هاي A باشد. ما اين رلس ها را در جهت برعكس پردازش مي كنيم. در هنگام پردازش هر رلس X از A ما دو مقدار براي آن تعريف مي كنيم: S_X، $\sum^*(X)$. براي حل مساله با شروع از Y_m و قرار دادن $S_{Y_m}=0$ و $\sum^*(Y_m)$ همه كارگرها را به $\perp$ منسوب مي كنيم. حال براي $1 \le h < m$ نشان مي دهيم چگونه S_{Y_h} و $\sum^*(Y_h)$ تعيين مي شوند. فرض كنيم $D_0, D_1, \ldots, D_k$ تمام زير مجموعه هاي مجزاي $E(Y_h)$ باشند. به‌طوري كه $\phi = D_0$. حال تمام انتساب هاي ممكن كه برابر $(E(Y_h)+1)^n$ هستند را براي n كارگر در نظر مي گريم. (هر كارگر يا به $E(Y_n)$ يا $\perp$ منسوب مي شود) براي هر انتساب احتمال اينكه D_i مجموعه كارهاي انجام شده به ازاي انتساب باشند را محاسبه مي كنيم. در صورتي كه $0 < a_1+a_2+\ldots+a_k$ باشد جمع وزندار $\frac{1}{a_1+a_2+\ldots+a_k}(1+\sum_{i=1}^{k} a_i(Y_h \cup D_i))$ را محاسبه مي كنيم. آنگاه آن انتسابي كه اين جمع به ازاي آن كمينه است را به عنوان $\sum^*(Y_h)$ در نظر مي گريم و مقدار S_{Y_h} را برابر اين جمع قرار مي دهيم. حال با كاهش h بر روي ترتيب مكاني حركت مي كنيم. هنگامي كه تمام دنباله پردازش شد Σ^* تعيين مي‌شود.

[1] Backtrack

۴-۲ تحلیل درستی الگوریتم

در این بخش نشان داده می شود که انتساب بهینه Σ^* را مي توان در زمان چند جمله اي محاسبه کرد. قضيه بعدي نشان مي دهد که برنامه نويسي پويا، Σ^* بهينه را پيدا مي كند.

قضيه ٤: *پس از اجراي الگوريتم OPT*

- *Σ^* بهينه براي هر راس x از گراف A تعيين شده*
- *زمان اتمام انتساب با شروع از X محاسبه شده است و اين زمان کمترين زماني است که هر انتساب با شروع از X مي تواند دست يابد.*

اثبات: فرض کنيد $Y_1,Y_2,...,Y_m$ ، ترتيب مکاني از A که توسط الگوريتم استفاده مي شود. فرض کنيد يك متغير است به طوري که براي هر i؛ $h \le i \le m$، $S_{Y_i} = B_{Y_i}$ و S_{Y_i} برابر زمان تکميل انتساب Σ^* با شروع از Y_i است.

براي h=m اين گزاره بدون شك درست است. در واقع، با توجه به لم ۲، A تنها يك چاه دارد که برابر N_G است، بنابراين $Y_m=N_G$ اگر همه کارها انجام نشده باشند، زمان تکميل است. پس بنابراين الگوريتم به درستي S_{ym} را برابر مي گذارد. زمان تکميل هر انتسابي با شروع از حالت N_G برابر است بنابراين $\sum^*(Y_m)$ يك انتساب بهينه است.

حل $2 \le h \le m$ در نظر مي گيريم. بايد ببينيم وقتي الگوريتم Y_{h-1} را بررسي مي كند گزاره با كاهش h به اندازه ١ واحد درس است. از آنجايي كه x در بردارنده همه چاه ها نيست. حداقل يك فرزند دارد. فرض كنيد $x_1, x_2, \ldots, x_k$ فرزندان X باشند.

ابتدا نشان داده مي‌شود كه X حداكثر مقدار S_{Y_x} و B_x است. الگوريتم از بين تمام انتساب هاي ممكن انتسابي را انتخاب مي كند كه با شروع از x كه S_{Y_x} را كمينه مي كند. لم ١-٢ نشان مي دهد كه B_x محدود است. با توجه به قضيه ٢ داريم:

$$B_X = \frac{1}{a_1 + \ldots + a_k}(1 + \sum_{i=1}^{k} a_i T_{X_i})$$

T_{xi} زمان تكميل براي انتساب Σ باشروع از x_i است. از آنجايي كه $S_{xi} = B_{xi}$ و حداقل مقدار T_{xi} برابر S_{xi} است برنامه ريزي پويا همه انتساب هاي ممكن را در نظر مي گيرد و براي هر انتساب Σ ، جمع وزندار $\frac{1}{a_1 + a_2 + \ldots + a_k}(1 + \sum_{i=1}^{k} a_i S_{X_i})$ را حساب مي كند. از آنجايي كه جمع وزندار حداكثر B_x است و الگوريتم جمعي را انتخاب مي كند كه مقدارش كمينه باشد. پس حداكثر مقدار S_x برابر B_x است.

هنگامي كه مقدار Σ^* و S_x تعيين شدند، S_x به‌طور واضح برابر زمان تكميل Σ^* باشروع از X است. در واقع براي هر انتساب Σ^*، جمع وزندار برابر زمان تكميل Σ^* باشروع از x يا در واقع $\Sigma^*(X)$ است.

از آنجايي كه Σ^* زمان تكميل S_X و S_X حداكثر برابر B_X است. بنابراين زمان تكميل Σ^* با شورع از X در واقع برابر B_x است. از آنجايي كه اثبات براي كاهش به اندازه ۱ واحد h است پس گزاره در حالت كلي درست است.

۵-۲ تحليل زمان اجراي الگوريتم

حال نشان می شود كه مي توان در زمان چند جمله اي براي گراف جهتدار بدون دوري كه عرض آن ثابت و تعداد كارگران ثابت باشد،مسئله را حل كرد.

فرض كنيد ω عرض گراف است. با توجه به قضيه ديلورس مي توان گراف را با ω زنجير پوشش داد. براي اينكه كار I در زنجير انجام شود بايد تمام كارهاي و زنجير كه قبل از كار Iام هستند انجام شده باشند. حداكثر طول هر زنجير برابر t است. پس تعداد مجموعه هاي مجزا كه شرط پيش نيازي را دارند برابر $(t+1)^{\omega}$ است.

وقتي كه الگوريتم هر راس A را پردازشمي كند. الگوريتم كليه انتساب هاي ممكن كارگران به كارها را در نظر مي گيرد. هر كارگر حداكثر به t+1 كار مي تواند منسوب شود و بنابراين حداكثر تعداد انتساب ها برابر $(t+1)^{n}$ است.

براي هر انتساب تعداد a_iهايي كه حداكثر بايد محاسبه شوند 2^n مجموعه D_i ممكن است. چرا كه كارگرها حداكثر به n كار مختلف منسوب مي شوند. بنابراين احتمال دستيابي كه مجموعه هايي كه

زیر مجموعه کارهاي منسوب شدن نیستند برابر است باید از جمع وزندار حذف شوند. بدین صورت قضیه بعدي به دست مي آید.

¤

قضیه 5 *: یك الگوریتم با زمان اجراي چند جمله اي براي حل مساله* ROPAS *هنگامي كه عرض گراف و تعداد كارگرها ثابت باشد. مثالي از كاربرد الگوریتم در شكل 1 آمده است.*

۳مساله برنامه ریزی کارگران غیر قابل اعتماد با زنجیره مارکوف

۳ - ۱ تعریف هاي پایه

۳-۱-۱ زنجیره مارکوف زمان گسسته ۱ (DTMC)

یك زنجیر مارکوف زمان گسسته یك سه تايي $(S, P, \bar{S})$ است كه S مجموعه محدود از حالت[2] هاي ممكن است، P يك ماتريس احتمال است، $P: S \times S \to [0,1]$، به طوري كه $\sum_{s' \in S} P(s,s') = 1$ براي تمامي $s \in S$ و $\bar{S}$ حالت شروع است. هر، $P(S, \bar{S}) > 0$ احتمال انتقال از S به $\bar{S}$ است. يك مسير در DTMC يك دنباله از حالت ها $S_0\ S_1\ \ldots\ S_k$ است كه $s_0 = \bar{s}$ و $P(s_i, s_{i+1}) > 0$ براي هر $0 \le i < k$.

درمساله تعريف‌شده حالت خاصي از DTMC استفاده‌شده است. در DTMC استفاده‌شده از تمامي حالت ها دقيقاً دو انتقال[3] امكان پذير است. يعني بر هر حالت S،ما تنها دو حالت 'S داريم كه $P(s,s') > 0$. يك يا هر دوي اين حالت ها ممكن است s باشند. اين دو انتقال نشان دهنده رفتار يك

---

[1] Discrete time markov chain

[2] State

[3] Transfer

کارگر وقتي در حالت s است، مي باشند. يکي از اين انتقال ها «موفقيت»[1] و ديگري «شکست»[2] برچسب گذاري[3] مي‌شوند. اگر کارگر در انجام يك کار موفق شود به حالت بعدي که توسط موفقيت تعيين مي‌شود مي رود و در غير اين صورت به حالت شکست مي رود.

۳-۱-۲ کارگران مارکوفی

مساله يك گسترشي از مساله از[1] است. در مساله ROPAS، n کارگر براي انجام t کار داريم. کارها {1,2,…,t} يا با [t] نمايش مي دهيم. به همين ترتيب مجموعه کارگران {1,2,…,n} يا [n] است.

کارگر i کار j را با احتمال $P_{i,j}$ به درستي انجام ميدهد. هدف يافتن برنامه ريزي است که زمان مورد انتظار براي تکميل کار را کمينه کند.

ماساختاري را در نظر مي گريم که کار j مي تواند هر کارگر i منسوب شود. هر کارگر يك واحد زماني براي انجام کاري که به او منسوب شده مي کند. هر کارگر يك DTMC، M_i هم دارد و

[1] Success

[2] Fail

[3] Label

رفتار آن به این DTMC ارتباط دارد. در ابتدا کارگر i در حالت $\bar{s}$ از M_i است. بعد از آن بین حالت ها با توجه به نام انتقال، حرکت مي كند.

هنگامي که کارگر i براي انجام کار زمنسوب شده در حالت s_k تلاش مي كند، و احتمال او را همراهي مي كند:

۱- $P_{i,j}$ كه $0 \le P_{i,j\le 1}$ يك احتمال ثابت درطول زمان است و از ابتدا مشخص است.

۲- $0 < q_{s_k} \le 1$ كه احتمال موفقيت در حالت s_k است.

هدف یافتن انتساب Σ که زمان تکمیل کمینه براي تکمیل تمامي كارها دارد. در هر بازه زماني، تعدادي از كارها براي انجام مجاز هستند. انتساب تعدادي از اين كارها را به تعدادي از كارگرها منسوب مي كند. ممكن است تعدادي از كارگران بيكار باشند و يا تعدادي از كارها به كسي منسوب نشوند. بعد از يك واحد زماني تعدادي از كارها (یاشاید هیچ کاري) انجام شوند. فرايند همين كار را ادامه مي دهد تا تمامي كارها انجام شوند.

۳-۱-۳ گراف ماركوف:

گراف ماركوف G_M، گرافي است كه به ازاي هر حالت s_i يك رلس در گراف داريم. هر رلس s_i دقيقاً دو يال خروجي دارد كه به رلس هاي s_{i1} و s_{i2} مي رويد (ممكن است يكي از s_{i1} و s_{i2} يا هر دو برابر s_i باشند). يكي از اين يال ها وزن $0 < q_{s_k} \le 1$ و ديگري وزن $1 - q_{s_i}$ دارد. يال داراي وزن

q_{si} را يال موفقيت و يال ديگر را يال شكست برچسب گذاري مي كنيم. براي اين گراف دو تابع تعريف مي كنيم:

۱- تابع $success: N_{G_M} \rightarrow N_{G_M}$ به ازاي رلس s_i درصورت پيمايش يال موفقيت به چه رلسي مي رسيم.

۲- تابع $Fail: N_{G_M} \rightarrow N_{G_M}$: به ازاي رلس s_i درصورت پيمايش يال شكست به چه رلسي مي رسيم.

هر كارگر i، درمساله داراي يك گراف ماركوف است كه با G_{Mi} نشان داده مي‌شود.

۳-۱-۴ گراف با درجه خروجي ۱:

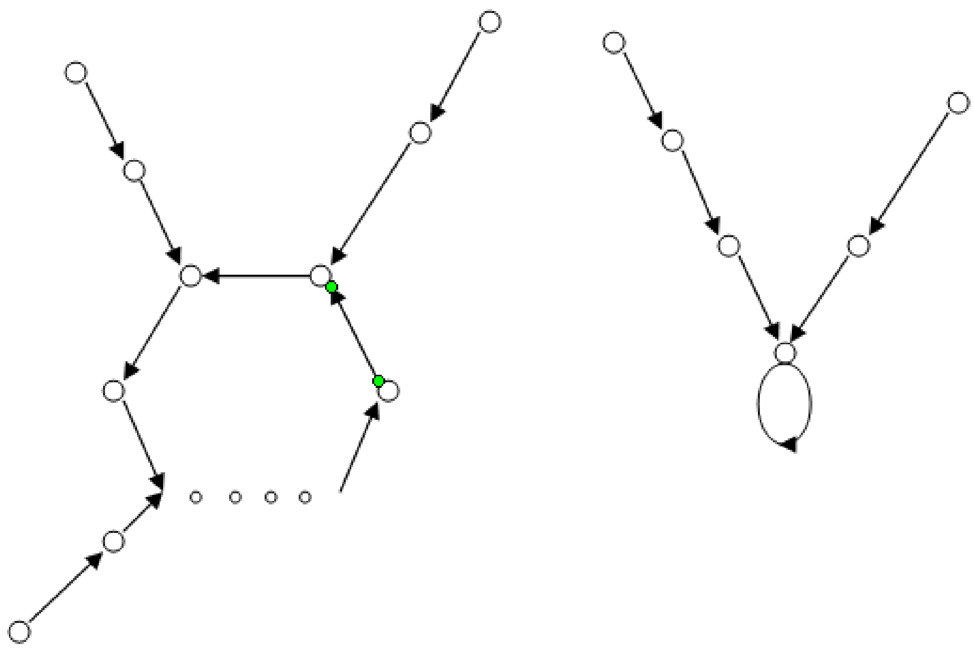
فرض كنيد گراف جهتدار G(N,E) كه درجه خروجي هر رلس در آن دقيقاً يك است: اين گراف خصوصيات زير را دارد:

۱- |N| = |E|

۲- شكل گراف به‌صورت ازمولفه هاي همبندي است كه درهرمولفه دقيقاً يك دور داريم و تعدادمسير كه به اين دو ختم مي‌شوند.

يك جنگل رادرنظر بگريد. گراف n رلس و n يال مانند جنگلي كه به هر درخت آن يك يال اضافه كنيم (چرا كه در هرمولفه n' رلس و n' يال داريم). پس يك دور و تعدادمسير به اين دور داريم.

هر رئس که در دور است یك کمان ورودي و یك کمان خروجي دارد پس رئس هایي که در محل اتصال مسير به دور هستند حتماً کمان به آنها وارد مي شود و حال چون آخرين رئس مسير درجه خروجي است ۱ است یال قبلي مسير هم به آن واردمي‌شود و در نتيجه‌شكل يك‌سري‌مسير است كه به دور واردمي‌شود (شكل ۳)



شکل 8- نمونه ای از گراف با درجه ورودی یک

۳-۱-۵ گراف‌شكست كارگر

گرافی که از القاي يالي روي يال هاي‌شكست گراف ماركوف به دست مي آيد،گراف‌شكست كارگر است (G_{Fi}) .

این گراف $N_{G_{F_i}} = N_{G_{M_i}}$ و $E_{G_{F_i}} \subseteq E_{G_{M_i}}$ به‌طوري كه به ازاي هر كمان $u \to v$ در گراف شكست، برچسب $n \to v$ در گراف ماركوف آن شكست است.

این گراف یك گراف با درجه خروجي ۱ است.

۳-۱-۶ گراف شكست عمومي

گراف G_F گرافي است كه به ازاي هر دنباله ممكن از حالت هاي كارگران يك رلس دارد و در نتيجه:

$$|N_{G_F}| = |N_{G_{M_1}}| * |N_{G_{M_2}}| * ... * |N_{G_{M_n}}|$$

و هر رلس يك برچسب دارد كه دنباله متناظر آن را نشان مي دهد كه يك دنباله به‌طول n است.

$s_1, s_2, ..., s_n$ نشان ميدهد حالت كارگر i، s_i است.

بين رلس v_1 با دنباله $s_1, s_2, ..., s_n$ و رلس v_2 با دنباله $s'_1, s'_2, ..., s'_n$ يال داريم اگر و تنها اگر به ازاي هر i، $s'_i = Fail(s_i)$ در گراف G_{Mi} و از آنجايي كه هر رلس در G_{Mi}، حالت شكست آن يكتاست هر رلس در گراف شكست دقيقاً يك خروجي دارد.

روي گراف شكست عمومي تابع Next را بدين‌صورت تعريف مي كنيم.

$$Next : N_{G_{M_1}} \times N_{G_{M_2}} \times ... \times N_{G_{M_n}} \to N_{G_{M_1}} \times N_{G_{M_2}} \times ... \times N_{G_{M_n}}$$

که در واقع یك دنباله از حالت کارگران را مي گیرد $(s_1,s_2,\ldots,s_n)$ و دنباله $s_1',s_2',\ldots,s_n'$ از حالت کارگران را برمي گرداند که به ازاي هر i $s_i'=Fail(s_i)$.

۳-۱-۷ خصوصیات گراف شکست عمومي

۱- این گراف دقیقاً $|N_{G_F}|=|N_{G_{M1}}|\times\ldots\times|N_{G_{Mn}}|$ راس دارد.

۲- این گراف، یك گراف با درجه خروجي ۱ است.

دور $s_1,s_2,\ldots,s_L$ در این گراف را در نظر بگیرید (کلیه s_iها متفاوتند به جز s_1 و s_l):

$$S_1=(s_{11},s_{12},\ldots,s_{1n})$$

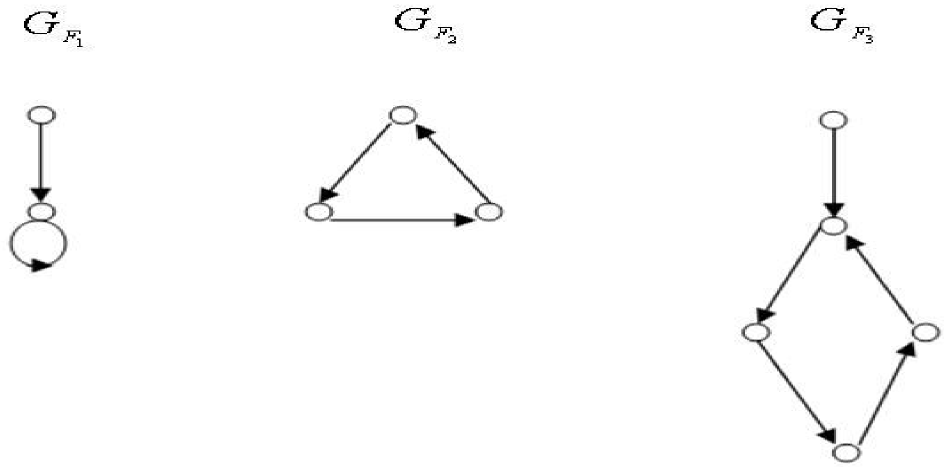
$$S_2=(s_{21},s_{22},\ldots,s_{2n})$$

$$\vdots$$

$$S_l=(s_{l1},s_{l2},\ldots,s_{\ln})$$

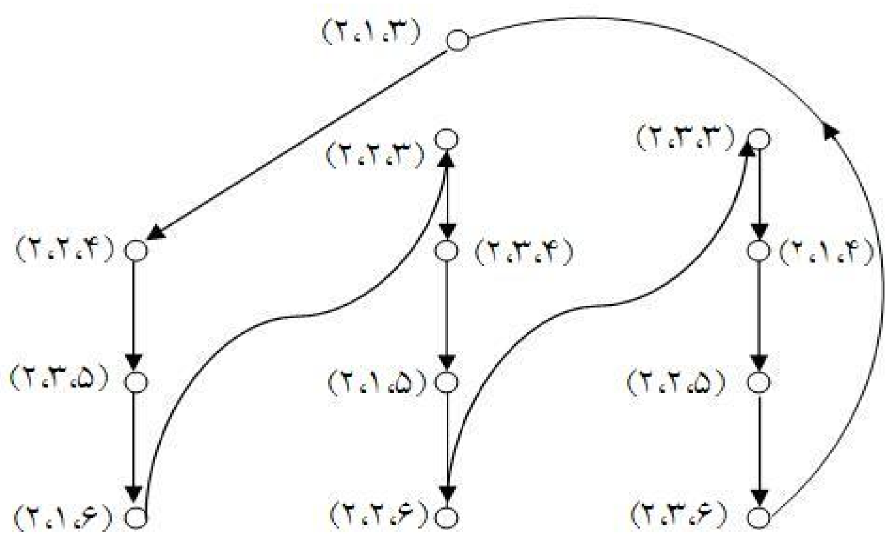
$$\forall\ 1\le i\le n \quad : \ s_{1i}=s_{li}$$

از آنجایي که $s_{li}=s_{1i}$ پس در گراف شکست کارگر i، $s_{11},s_{21},\ldots,s_{l1}$ یك گشت بسته هستند و در واقع مي توان نتیجه گرفت هر دور از یك گراف شکست کارگر در اثر ترکیب با دورهاي دیگر در گراف هاي شکست کارگر دیگر یك دور در گراف شکست عمومي مي‌سازد و در واقع طول دور در گراف شکست عمومي ک.م.م طول دورهاي تشکیل دهنده آن در گراف هاي مختلف است. به عنوان مثال گراف شکست ۳ کارگر اگر به صورت زیر باشد:



شکل9 **-گراف شکست سه کارگر**

دور موجود در گراف شکست عمومي به صورت



شکل 10 **-گراف شکست عمومی این سه کارگر**

خواهد بود که طول آن 12=(2,4,6) ک.م.م است. از آنجايي که حداکثر طول هر دو در هر گراف شکست کاگر $|N_{GMi}|$ است پس طول دور در گراف شکست عمومي $|M_i|^n$ است.

۳-۲ تعريف مساله

مساله يک گراف که بيانگر وابستگي کارها و n کارگر به‌طوري کارگر i با احتمال $0 \le P_{i,j} \le 1$ کار j را به درستي انجام مي دهد و براي هر کار j يک کارگر i وجود دارد که $0 < P_{i,j}$. به علاوه هر کارگر يک DTMC، M_i دارد که متشکل از حالت هايي و انتقال بين آنها است و هر حالت s_k، يک $0 < q_{s_k} < 1$ دارد که احتمال انتقال توسط موفقيت را در اين حالت نشان مي دهد.

هدف پيدا کردن برنامه ريزي بهينه Σ^* که «زمان تکميل» را کمينه کند.

در ابتدا لمي است که نشان مي دهد که زمان اتمام تمامي انتساب که با $B_{x,s}$ نشان داده مي‌شود در صورت شروع از مجموعه کارهاي انجام شده X و بودن کارگران در حالت هاي $S=(s_{1,0},s_{2,0},...s_{n,0})$ که $s_{i,0}$ حالت کارگران و زمان شروع انتساب است، محدود است.

لم ۱. *براي هر مجموعه x که شرط پيش نيازي را داشته باشد و براي هر حالت شروع اوليه کارگرها $B_{x,s}$ متناهي است.*

اثبات: اثبات ارائه شده بسيار شبيه اثبات بخش قبل است. تلاش مي‌شود که انتسابي مشخص شود که داراي زمان تکميل متناهي باشد. بنابراين $B_{x,s}$ نيز بايد متناهي باشد. انتساب بسيار ساده است:

فرض كنيد $t_1,t_2,...,t_m$ ترتيب ترتيب مكاني كارها باشد. كارها مي توانند بدين ترتيب انجام شوند چرا كه هنگامي كه ترتيب كارها در X، $t_1,t_2,...,t_{j-1}$ باشند، كار t_j براي انجام‌شدن مجاز است.

انتساب اين ترتيب را در نظر مي گيريد وقتي يك كار بايد انجام‌شود تمامي كارگرها به اين كار منسوب مي‌شوند. هر كارگر دو پارامتر براي احتمال موفقيت در انجام كار دارد. فرض كنيد كارگر در حالت $s_{i,j}$ باشد وقتي كه مي خواهد كار ژام را انجام دهد و بنابراين احتمال موفقيتش در اين حالت $q_{s_{i,j}}$ است. ازطرفي احتمال انجام كار t_j توسط كارگر i، $P_{i,j}$ است. بنابراين كارگر i، اين كار را با احتمال $P_{i,j} \times q_{s_{i,j}}$ به درستي انجام مي دهد و بنابراين احتمال شكست براي انجام t_j برابر $\prod_{i=1}^{n}(1-P_{i,j}q_{s_{i,j}})$ است. درصورتي كه كار با موفقيت انجام نشود، دنباله حالت كارگران برابر $s_1',s_2',...,s_n' = Next(s_1,s_2,...,s_n)$ خواهدشد و به همين ترتيب سناريو ادامه مي يابد. از آنجايي كه تمام q_{si}ها براي تمام كارگران بزرگتر از صفر است مي توان q_{min} را بين تمامي اين q_{si}ها در تمامي M_iها تعيين كرد و در هر حالتي از كارگران كه باشيم براي انجام كار t_j:

$$\prod_{i=1}^{n}(1-P_{i,j}q_{s_{i,j}}) \le \prod_{i=1}^{n}(1-P_{i,j}q_{\min})$$

از آنجايي كه $0<q_{min}$ و به ازاي حداقل يك كارگر $0 \neq P_{i,j}$ است پس اين ضرب از يك كوچكتر است و پس از $\frac{1}{1-\prod_{i=1}^{n}(1-P_{i,j}q_{\min})}$ مرحله اين كار حتماً انجام مي‌شود (اين نتيجه با توجه به اينكه زمان اجرا توزيع هندسي دارد به دست مي آيد).

از نتیجه به دست آمده در بالا مي توان گفت که $B_{X,S} \leq \sum_{j=1}^{t} \frac{1}{1-\prod_{i=1}^{n}(1-P_{i,j}q_{\min})}$ که مقدار متناهي دارد.

¤

براي محاسبه $B_{x,s}$ما يك رابطه بين $B_{x,s}$ و $B_{x',s'}$ براي هر مجموعه انجام‌شده از کارها X و هر حالت کارگران S به دست مي آوريم.

قضيه ۳ *: انتساب Σ، مجموعه x که شرط پيش نيازي را دارد و تمامي چاله هاي G در آن قرار ندارد و حالت کارگران $S=(s_1,s_2,...,s_n)$. فرض زمان مورد انتظار انتساب Σ براي کارهاي انجام‌شده X و حالت S، متناهي باشد. فرض کنيد $D_0,D_1,...,D_k$ کليه زير مجموعه هاي مجزاي E(x) باشند، $D_0=\phi$. فرض کنيد Next(s)، $S_0,S_1,...,S_l$ کليه حالت هاي بعدي ممکن براي کارگران باشد و Allfail(S)=S_0. فرض کنيد $a_{i,j}$ احتمال اينکه D_i دقيقاً مجموعه اي از کارها باشد که با توجه به انتساب Σ توسط کارگران انجام مي‌شود و S_j حالت بعدي کارگران باشد. $X_i = X \cup D_i$ قرار دهيد. آنگاه:*

$$T_{X_0,S} = 1 + \sum_{i=0}^{k}\sum_{i=0}^{l} a_{i,j} T_{X_i,S_j}$$

درصورتي که T_{X_i,S_j} زمان مورد انتظار براي تکميل انتساب Σ با شروع از کارهاي انجام‌شده‌ي x_i و حالت کارگران S_i باشد.

اثبات: براي اثبات قضيه به ازاي هر جفت (X_i,S_j) را يك حالت در نظر مي گيريم كه كارهاي انجام شده X_i و حالت كارگران S_j براي تمام حالت ممكن انجام كارها توسط كارگران، حالت (X'_i,S'_j)، يك انتقال، $(X_i,S_j) \to (X'_i,S'_j)$، است. احتمال مي توان به آساني با جمع وزن تمام حالت هاي ممكن انجام كارها كه به X'_i و حالت كارگران S'_j ختم مي‌شود انجام شود.

$T_{X_0,S}$ را در نظر بگيريد اين زمان از دو بازه تشكيل شده است. بازه اول كه برابر ۱ واحد زماني است كارگران سعي مي كنند كارهايي كه به آنها منسوب شده است، را انجام دهند. بعد از يك واحد زماني مشخص مي‌شود كه هر كارگر در انجام كار منسوب شده موفق شده است يا نه.

تعدادي از كارها (يا هيچ كدام) به درستي انجام مي‌شوند و حالت بعدي كارگران با توجه به عملكرد آنها تعيين مي شود. بعد از اين كار با S,X جديد ادامه مي يابد. فرض كنيد ما به ازاي S,Xهاي جديد T را محاسبه كرده ايم. بنابراين $T_{X0,S}$ جمع وزندار كليه حالت هاي ممكن بعدي است و با جمع اين دو بازه زماني خواهيم داشت:

$$T_{X_0,S} = 1 + \sum_{i=0}^{k}\sum_{j=0}^{l} a_{i,j} T_{X_i,S_j}$$

¤

ممكن از (X_0,S_0) به خودش انتقالي وجود داشته باشد در اين صورت مي توان معادله بالا را به صورت زير بازنويسي كرد:

$$T_{X_0,S} = \frac{1}{\sum_{i=0}^{k} \sum_{i=0,i+j\neq 0}^{l} a_{i,j}} \times (1 + \sum_{i=0}^{k} \sum_{i=0,i+j\neq 0}^{l} T_{X_i,s_j})$$

وابستگي بين جفت (X,S)هاي مختلف را به صورت يك گراف مدل مي كنيم. گراف جهتدار $A=(N_A,A_A)$. گراف تكامل مجاز ماركوفي از اجراي كارهاي G را بدين صورت مي سازيم(مثال در شكل ۳):

گراف A را به‌صورت استقرايي مي‌سازيم. هر رلس از A نشان دهنده رئوسي از G و حالتي از حالت كارگران است.

ابتدا كه $\bar{S} = (\bar{s}_1, \bar{s}_2, ..., \bar{s}_n)$ كه $\bar{s_i}$ حالت اوليه هر كارگر است. در هر مرحله به ازاي هر رلس از $(X,S) \in N_A$ كه Xشامل كليه چاه هاي G نمي باشد:

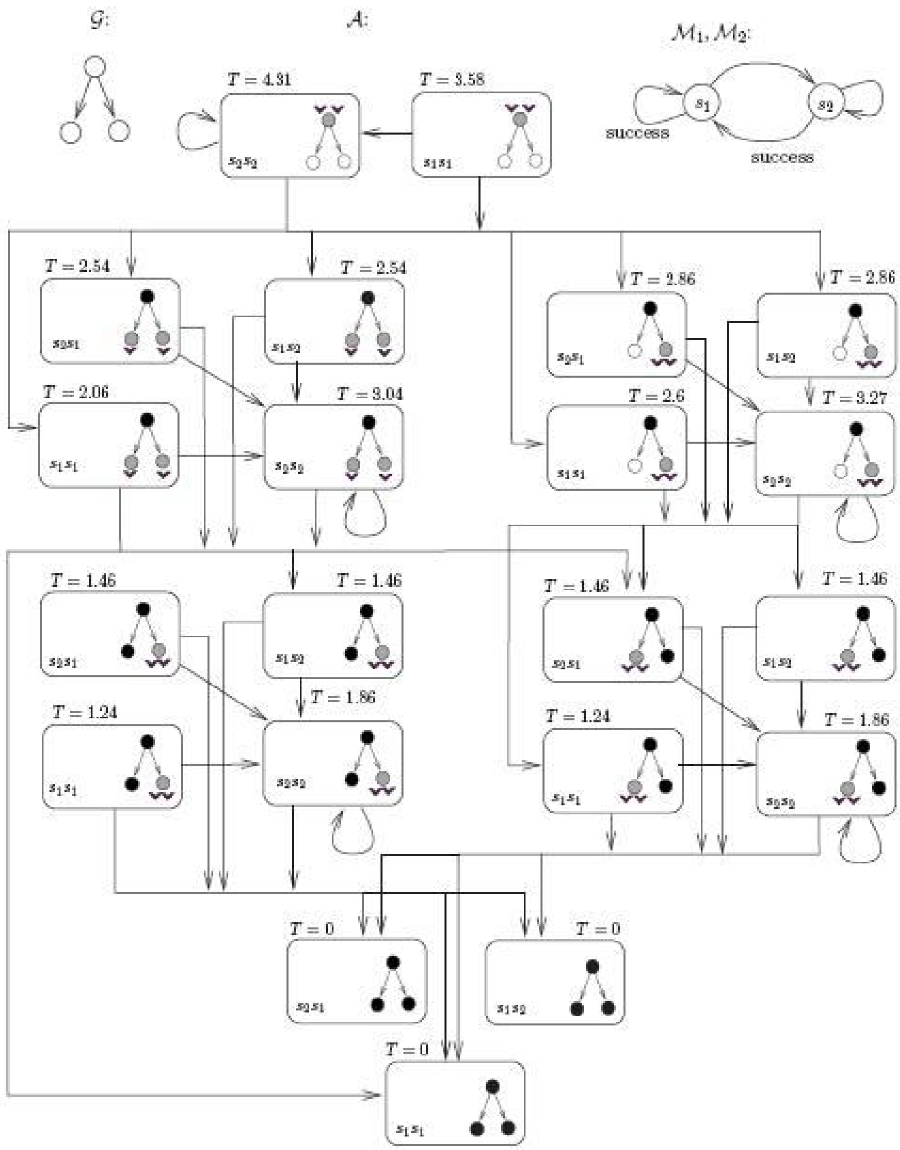
۱) E(x) رامحاسبه مي كنيم. حال تمامي زيرمجموعه هاي غيرتهي $D \in E(x)$ را درنظر مي گيريم.

۲) فرض كنيم $S=(s_1,s_2,...,s_n)$ كليه هاي دنباله هاي ممكن 's، كه زيرمجموعه‌ي

$$\{success(s_1), fail(s_1)\} \times \{success(s_2), fail(s_2)\} \times ... \times \{success(s_n), fail(s_n)\}$$

هستند را درنظر مي گيريم.

به ازاي كليه جفت هاي $(X \cup D, S')$ ، رئوس $(X \cup D, S')$ را درصورتي كه درمجموعه N_A نباشند به آن اضافه مي كنيم و كمان $((X,S),(X \cup D,S'))$ را به A_A اضافه مي كنيم.



شكل ۱۱ مثالي از گراف تكامل ماركوفي

در گراف تکامل مجاز مارکوفي يا به‌طور ساده تر گراف وابستگي، تنها دورهاي ممکن طوقه ها و دورهاي بين راس هاي با X يکسان است و در واقع در اينجا دورها متناظر دورها در گراف شکست هستند چرا که X که ثابت است و يال بين xها يالي است که کليه کارگران شکست بخورند و در نتيجه دقيقاً بدون در نظر گرفتن x همان گراف شکست را داريم و در واقع انگار به ازاي هر x يك کپي از گراف شکست داريم.

گراف شکست عمومي بدون دور G'_F را از روي گراف شکست عمومي G_F بدين صورت مي سازيم که به ازاي هر دور در G_F يك يال از آن دور حذف مي کنيم و در نتيجه G'_F بدون دور خواهد بود.

فرض کنيد $S_1, S_2, \ldots, S_l$ ترتيب ترتيب مکاني رئوس G'_F باشند.

گفتيم که به ازاي هر X از رئوس A يك کپي از گراف شکست عمومي داريم. حال در واقع به ازاي هر X يك کپي از 'A را قرار مي دهيم. (يعني يك يال از دورها حذف مي کنيم) حال ترتيب مکانی رئوس A را که به‌صورت

$$(\varnothing,S_1),(\varnothing,S_2),\ldots,(\varnothing,S_L),$$

$$(X_1,S_1),(X_1,S_2),\ldots,(X_1,S_L),$$

$$\vdots$$

$$(N_G,S_1),(N_G,S_2),\ldots,(N_G,S_2)$$

خواهد بود. اگر این دنباله را با $Y_1,Y_2,\ldots,Y_m$ نشان دهیم، هر یک از Y_i ها در واقع دنباله ای از جفت های (X_i,S_j)، $1 \le j \le l$، است.

۳-۳ الگوریتم مساله کارگران مارکوفی

با فرض اینکه عرض گراف و تعداد کارگران ثابت است، بعلاوه طول دورها در گراف شکست کارگران ثابت است، براي مسئله الگوريتمي ارائه مي دهيم (با توجه به اینکه طول دور در گراف شکست عمومي هم ثابت چرا که حداکثر c^n خواهد بود و چون c و n ثابت پس ثابت است).

الگوریتم ارائه شده در واقع شبیه الگوریتم ارائه شده براي مسئله بدون داشتن زنجیره مارکوف است. براي هر رلس یک $S_{X,S}$ که زمان تکمیل مورد انتظار آن رلس و یک $\sum^*(X,S)$ که انتسابي که باعث دستیابي به این زمان تکمیل میشود را داریم.

فرض کنید ترتیب مکانی رئوس گراف بدون دور 'A همان طور که در بخش قبل توصیف شد $Y_1,Y_2,\ldots,Y_m$ باشد. حال از آخر به اول شروع به پیمایش این دنباله ميکنیم. ابتدا براي تمام اعضاي Y_m، $S_{X_m,S_i}=0$ و $\sum^*(X_m,S_i)$ همه کارگران را به $\perp$ منسوب ميکند. حال براي $1 \le h \le m$ نشان ميدهیم که چگونه S_{X_h,S_i} و $\sum^*(X_h,S_i)$ تعيين ميشود:

$|E_{x_h}| \le w$ است زیرا بین هیچ دو رلسي از آن نميتوان مسیر داشت چون در اینصورت پیشنیازهاي یکي به‌صورت کامل انجام نشده‌است. حداکثر تعداد انتساب‌هاي ممکن براي کارگران $(w+1)^n$ است (هر کارگر یا به $E(X_h)$ و یا به $\perp$ منسوب ميشود).

هر کارگر در انجام کارش موفق یا ناموفق خواهدبود پس 2^n سناریوي مختلف براي انجام کارها داريم، به ازاي هرسناريو مي‌توان حالت بعدي کارگران را با توجه به‌شکست يا موفقيت هرکدام به دست آورد و مجموعه کارهاي انجام‌شده نيز کارهاي مي‌شوند که حداقل يکي از کارگران منسوب شده براي انجام آن موفق شده باشد.

فرض کنيد که $D_0, D_1, ..., D_k$ کليه زيرمجموعه‌هاي $E(X_h)$ باشند $D_0 = \phi$. پس احتمال دستيابي به هر يک از $(X \cup D_i, S'_j)$ ها برابر جمع احتمال سناريو هايي که نتيجه رسيدن به اين جفت را دارند مي‌شوند. به ازاي هرسناريو C، که با يک دنباله به‌طول n از ۰ و ۱ که يک نشان موفقيت کارگر است:

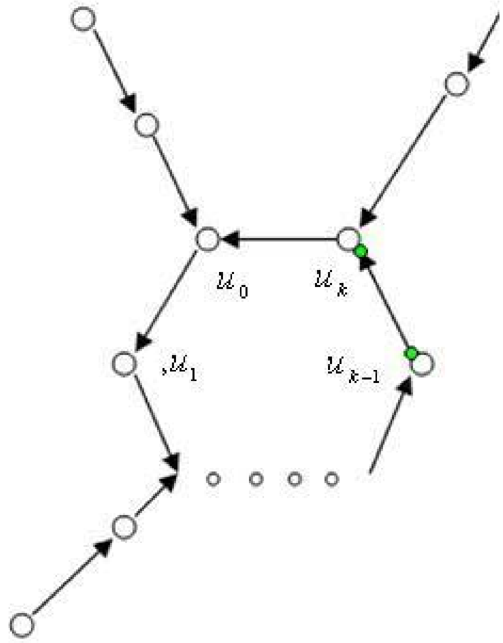
اگر حالت کارگران در حال حاضر S باشد درصورتي که t_l کار منسوب شده به کارگر l ام را نشان دهد.

$$a_{i',j'} = \sum_{C[l]=1} P_{l,t_l} * q_{l,s_l} + \sum_{C[l]=0} 1 - P_{l,t_l} * q_{l,s_l}$$

اگر $D \neq \phi$ باشد Sمربوط به آن راس قبلا حساب‌شده بنابراين تنها حالتي که بايد بررسي‌شود حالتي است که $D = \phi$ است و در واقع تفاوت اساسي مساله ماركوف در همين قسمت است.

با توجه به پيمايشي که براي گراف‌شکست کارگر با حذف يک دور داريم و چون ترتيب مکاني آن بود پس حتما (X_h, S') قبلا محاسبه شده مگر براي آن راسي که در دور بوده و يال وصل‌کننده آن به فرزندش حذف‌شده بوده که اين راس دقيقا آخرين راس ترتيب مکاني درمؤلفه همبندي خودش است.

چرا که با توجه به شکل مؤلفه که تعدادي مسير است که به یک دور ختم می‌شود.

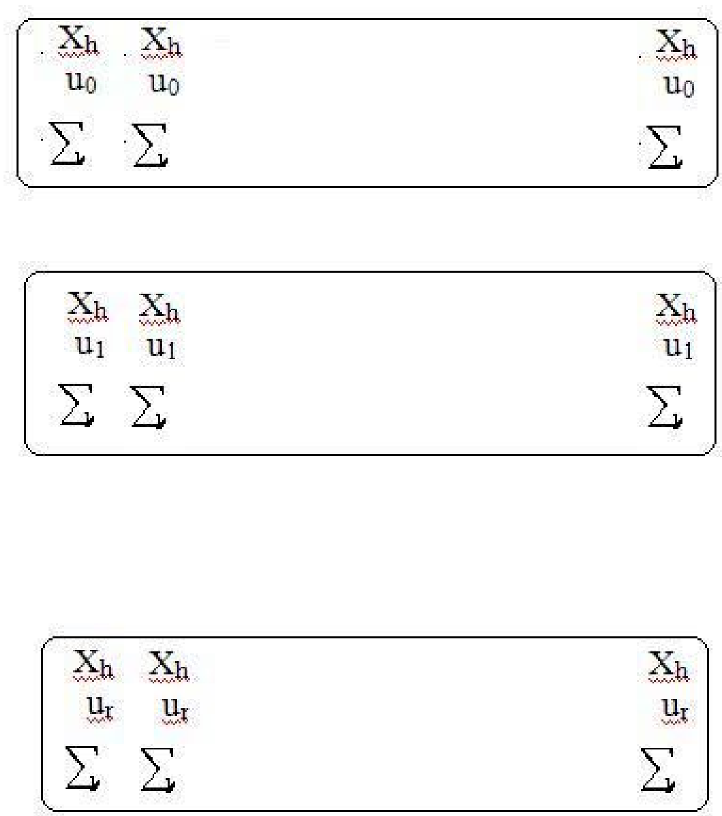


شکل 12- شمای یک مولفه همبندی از گراف شکست عمومی

حال اگر يال $u_0 \longrightarrow u_k$ در دور حذف شده باشد رأس‌هاي دور به ترتيب مسير u_0 به u_k در ترتيب مکانی مي‌آيند و ساير رأس‌هايي که در دور نيستند چون در مسير به يکي از رأس‌هاي دور قرار دارند پس قبل از u_k در ترتيب مکانی مي‌آيند.

پس تنها رأسي که در محاسبه S آن شکل داريم رأس u_k است چرا که در حالتي که همه کارگران در انجام کارشان ناموفق باشند به جفت (X_h, u_0) مي‌رويم ولي S_{X_h,u_0} حساب نشده است. براي حل اين

مشكل وقتي به رأس (X_h,u_0) در محاسبات مي‌رسيم كه S آن حساب نشده به ازاي تمام انتساب‌هاي ممكن سعي كنيم كه S آن را حساب كنيم. حال چون S جفت (X_h,u_1) محاسبه نشده نمي‌توان S، X (X_h,u_0) را حساب كرد پس همين سناريو را براي (X_h,u_1) تكرار مي‌كنيم و آنقدر ادامه مي‌دهيم تا به (X_h,u_l) برسيم.



شكل ۷ 2نمونه از تعداد انتساب هاى ممكن

به ازاي كليه $((w+1)^n)^r$ معادله ممكن بدين طريق بدست مي‌آيند S_{X_h,u_0} را محاسبه مي‌كنيم و هر كدام مقدار S_{X_h,u_0} را كمينه كرد را انتخاب مي‌كنيم.

لم ۳: *كليه S_{X_h,u_i}، $1 \le i \le l$ هايي كه در معادله‌يي كه S_{X_h,u_0} كمينه در آن قرار دارند، كمينه هستند.*

اثبات: فرض كنيم S_{X_h,u_i} برابر مقدار كمينه S براي جفت (X_h,u_i) است. از آنجايي كه

$$S_{X_h,u_i} = P_i * S_{X_h,u_{i+1}} + C_i$$

برابر مقدار كمينه $S_{X_h,u_{i+1}}$ براي جفت (X_h,u_{i+1}) نباشد پس $S_{X_h,u_{i+1}}$ كمينه نيست پس S_{X_h,u_i} كمينه نيست. چون S_{X_h,u_0} كمينه است پس كليه S_{X_h,u_i} ها هم كمينه هستند.

چون S حاضر در جفت‌هاي (X_h,u_i) برابر مقدار كمينه شدند پس $\sum$ حاضر در آنها هم $\sum^*_{(X_h,u_i)}$ است.

مرحله ۲: محاسبه S , $\sum$ براي رأس‌هاي روي مسيرهاي منتهي به دورها.

فرض كنيم مسير $v_1,v_2,...,v_k$ را به دور داشته باشيم كه v_k رأس روي دور است پس S_{X_h,v_k} بهينه را محاسبه كرده‌ايم. حال روي دنباله از آخر به اول شروع به حركت مي‌كنيم و به ازاي كليه انتساب‌هاي ممكن S_{X_h,v_i} را با توجه به فرمول محاسبه مي‌كنيم. از آنجايي كه كليه S_{X_h,S_i} هايي كه در طرف راست معادله قرار دارند قبلا محاسبه شده‌اند اين كار امكانپذير است. پس براي هر رأس آن در S_{X_h,v_i} كه مقدارش كمينه است و $\sum$ مربوط به آن را در نظر مي‌گيريم.

۴-۳ تحليل زمان اجراي الگوريتم

فرض كنيد w عرض گراف باشد با توجه به قضيه Dilworth مي‌توان گراف را با $(t+1)^w$ زنجير پوشش داد (گراف تشكيل شده صرفا بر مبناي X ها). حال براي هر X با توجه به شكل گراف شكست عمومي ابتدا براي هر دور S , $\sum$ را محاسبه مي‌كرديم كه تعداد معادله‌هاي ممكن $((w+1)^n)^r$ (انتساب‌هاي ممكن براي دنباله به طول r) (r طول دور كه چون ك م م. دورهاي گراف شكست كارگران و با توجه به فرض ثابت است) و براي محاسبه P_i, C_i در هر معادله 2^n سناريوي عملكرد كارگران را داريم و براي رئوس جدا از دور صرفا $(w+1)^n$ انتساب ممكن و براي هركدام 2^n براي بدست آوردن $a_{i,j}$ داريم پس:

$$(t+1)^w * \left(\sum_{u\ in\ circle} ((w+1)^n)^r * 2^n + \sum_{u\ not\ in\ circle} (w+1)^n * 2^n\right)$$

تعداد كل اين رأس‌ها برابر XXX $\sum_{u\ in\ circle} ((w+1)^n)^r * 2^n$ و براي هركدام در زمان چندجمله‌اي تعداد قابل محاسبه است، پس در كل مي‌توان در زمان چندجمله‌اي محاسبه كرد.

۴ پیچیدگي مساله کارگران غیر قابل اعتماد

براي نشان دادن لازم بودن هر دو شرط ابتدا نشان داده مي‌شود که اگر عرض گراف ثابت و تعداد کارگران ثابت نباشد مساله سخت تمام است. به همین ترتیب در ادامه نشان داده مي‌شود که اگر عرض گراف غیر ثابت و تعداد کارگران ثابت باشد مساله سخت تمام است.

در حالتي که مساله دقیقا یک کار براي انجام دارد، جواب مساله بدیهي است چرا که تمامي کارگران را براي انجام این کار منسوب مي کنیم. وقتي که گراف تنها از دو کار مستقل تشکیل شده است و تعداد کارگران غیر ثابت است، مساله سخت تمام است (این دومین ساده ترین گراف است.). سختي مساله در این حالت از آنجا ناشي مي‌شود که نمي دانیم چگونه کارگران را به‌طور متوازن بین این دو کار تقسیم کنیم.

در حالتي که مساله دقیقا یک کارگر براي انجام کارها دارد نیز جواب مساله بدیهي است چرا که ترتیب مکانی کارها را در نظر مي گیریم و در هر مرحله کارگر را به اولین کار انجام نشده به ترتیب مکانی منسوب مي کنیم. وقتي که تعداد کارگران دو تا مي‌شود و عرض گراف غیر ثابت است، مساله سخت تمام مي‌شود. سختي مساله در این حالت از آنجا ناشي مي‌شود که پیچیدگي گراف باعث مي شود در هر مرحله که کاري انجام مي‌شود تعداد زیادي کار براي انجام مجاز شوند و ما نمي دانیم چگونه این دو کارگر را به این کارها منسوب کنیم.

٤-١ برنامه ريزي گراف هاي باريک با تعداد زيادي کارگر سخت تمام است.

در اين بخش نشان داده مي‌شود ROPAS در حالتي که دو کار مستقل و تعداد کارگران غير ثابت است، سخت تمام است. براي اين کار دو مساله کمکي معرفي مي‌شود و پيچيدگي آنها تعيين مي‌شود. اولين مساله افراز ضربي است.

افراز ضربي

نمونه : يک عدد n و اعداد $s_1,s_2,...,s_n \geq 2$.

مساله : آيا مجموعه P وجود دارد به‌طوري که $\prod_{i \notin P} s_i = \prod_{i \in P} s_i$؟

مي توان سعي کرد مساله معروف افراز جمعي را به اين مساله کاهش داد. ما مي توانيم از اين واقعيت استفاده کنيم که اگر داشته باشيم $\sum_{i \notin P} r_i = \sum_{i \in P} r_i$ آنگاه داريم $\prod_{i \notin P} 2^{r_i} = \prod_{i \in P} 2^{r_i}$. اما متاسفانه $s_i = 2^{r_i}$، فضاي نمايي نسبت به r_i دارد و صرفا از اينکه براي مساله ي $\sum_{i \notin P} s_i = \sum_{i \in P} s_i$ جواب چند جمله اي وجود ندارد، نمي توان نتيجه گرفت که براي مساله ي $\prod_{i \notin P} s_i = \prod_{i \in P} s_i$ نيز راه حل چند جمله اي وجود ندارد.

اين مساله ي فضاي اختصاص يافته را مي توان با يک کاهش زيرکانه ي ديگر از مساله پوشش دقيق توسط مجموعه هاي سه عضوي که اصل هاي پايه از نظريه اعداد دارد، حل کرد.

لم :مساله افراز ضربي سخت تمام است.

اثبات : براي اثبات كاهشي از مساله ي پوشش دقيق توسط مجموعه هاي سه عضوي به اين مساله را داريم.

يک نمونه از X3C كه مجموعه ي $C=\{S_1,S_2,\ldots,S_N\}$ از زيرمجموعه هاي سه عضوي از U={1,2,...,3m} است ، $S_i=\{a_i,b_i,c_i\}$.هر گاه زيرمجموعه اي از Cموجود باشد كه دقيقا هر عضو از U در دقيقا يک زيرمجموعه از S_i هاي منتخب از C اتفاق افتاده باشد.مي توان فرض كرد كه $m \le n$ است و اجتماع همه ي زيرمجموعه هاي C، U است چرا كه در غير اين صورت مساله جواب ندارد.يک نمونه از MP به شرح زيرمي سازيم :

كوچكترين اعداد اول متوالي را در نظر مي گيريم، $P_1,P_2,\ldots,P_{3m+1}$. با توجه به قضيه اعداد اول در منبع ۴، P_{3m+1}مقداري برابر $O(m\log m)$ دارد. بنابراين مي توانيم اين اعداد را در زمان $O(m^2\log^2 m)$ با استفاده از Erotosthenes sieve محاسبه كنيم. اين زمان با توجه به اندازه نمونه چند جمله اي است.

نمونه ي MP، اندازه اي برابر n+2 دارد. به هر عضو از C، عددي متناظر مي كنيم. اين عدد برابر حاصلضرب اعداد متناظر با عناصر اين زيرمجموعه هاست يعني $s_i=P_{a_i}P_{b_i}P_{c_i}\ge 2$. توجه كنيد s_i ها به اندازه ي كافي كوچک هستند، $O(n^6)$.فرض كنيد p ضرب s_i ها باشد و qضرب $P_1,P_2,\ldots,P_{3m}$ باشند كه هر كدام $O(n^{6n})$ هستند. توجه كنيد كه $\frac{p}{q}$ عدد صحيح است زيرا

$\bigcup_i S_i = U$. حل دو عدد دیگر به نمونه ي MP اضافه مي كنيم، $s_{n+1} = P_{3m+1}.\frac{p}{q}$ و

$s_{n+2} = P_{3m+1}.q$. بنابراين نمونه MP در زمان و حافظه چند جمله اي نسبت به n بدست مي آيد.

فرض كنيد كه X3C داراي اين جواب باشد و Q زيرمجموعه اي از C باشد كه جواب را معرفي مي كند. با توجه به تعريف مساله $\prod_{i \in Q} s_i = P_1 * P_2 * ... * P_{3m}$ است بنابراين $\prod_{i \notin Q} s_i = \frac{\prod_{i=1}^{n} s_i}{P_1 * P_2 * ... * P_{3m}}$ است.در واقع :

$$s_{n+1} \prod_{i \in Q} s_i = s_{n+2} \prod_{i \notin Q} s_i$$

بنابراين نمونه MP هم داراي جواب است.

فرض كنيد نمونه MP داراي جواب باشد و P زيرمجموعه اي باشد كه $\prod_{i \in P} s_i = \prod_{i \notin P} s_i$.مي دانيم كه تنها دو عدد s_{n+1} و s_{n+2} هستند كه بر P_{3m+1} بخشپذير هستند(P_{3m+1} را به عنوان فاكتور دارند.).مي دانيم دو عددطبيعي برابر هستند اگر و تنها اگر تجزيه آنها برابر باشد. پس n+1 با n+2 در P هستند.قرار دهيد {L=P\{n+1 و R را برابر عناصري كه در P نيستند به جز n+2 قرار دهيد. در نتيجه داريم :

$$s_{n+1} \prod_{i \in L} s_i = s_{n+2} \prod_{i \in R} s_i$$

كه مي تواند به صورت معادل زير نيز بيان شود :

$$\frac{p}{q}\prod_{i\in L} s_i = q\prod_{i\in R} s_i$$

که R و L افرازي از {1,2,...,n} هستند. بنابراين $p = \prod_{i\in L} s_i . \prod_{i\in R} s_i$ و با جايگذاري در معادله بالا به دست مي آوريم ، و $q^2 = (\prod_{i\in L} s_i)^2$ و از آنجايي که اين اعداد مثبت هستند داريم $q = \prod_{i\in L} s_i$. در نتيجه زير مجموعه اي از s_i موجود است که هر عنصر $P_1, P_2, \ldots, P_{3m}$ دقيقا در يکي از مجموعه ها آمده است و L زيرمجموعه اي از مجموعه ي C است که هر عنصر از U دقيقا در يک عنصر از L آمده است. پس نمونه X3C داراي جواب است.

¤

مسئله MP براي نشان دادن سخت تمام بودن دومين مسئله کمکي است.

افراز با کمينه جمع ضرب ها (*PMSP*)

نمونه : عدد n، اعداد گويا $0 < r_1, r_2, \ldots, r_n < 1$.

هدف : پيدا کردن مجموعه P که $\prod_{i\in P} r_i + \prod_{i\notin P} r_i$ کمينه مي باشد.

لم *: مسئله PMSP سخت تمام است.*

اثبات : یک کاهش تورینگ از MP به PMSP در زمان چندجمله اي ارائه مي‌شود. نشان داده مي‌شود که چگونه براي یک مساله ي MP با جستجو براي نمونه ي ساخته‌شده از PMSP، جواب بیابیم. اثبات بر این اصل بناشده است که بر ای هر d>0 در تابع $f(x) = e^{x} + e^{d-x}$ که بر روی $0 \le x \le d$ تعریف شده ، کمینه در $x = \frac{d}{2}$ اتفاق می افتد. تنها سختی حاضر در این جا کار با اعداد گویا است.

یک نمونه از مسئله MP را در نظر بگیرید که $s_1, s_2, ..., s_n$ همه اعداد صحیح بزرگتر از ۲ هستند. نمونه‌اي از PMSP با تعریف $r_i = h_i^{2}$ که $h_i = \frac{1}{s_i}$ در نظر مي‌گیریم. همان‌طور که مي‌خواستیم $0 < r_i < 1$ است. ما باید بررسي کنیم که هنگامي که براي مسئله MP جواب وجود دارد براي جواب کمینه براي نمونه ساخته‌شده از PMSP برابر $2\prod_{i=1}^{n} h_i$ است.

بررسي مي‌کنیم که چه زماني $\prod_{i \in P} r_i + \prod_{i \notin P} r_i$ این مقدار را دارد. $d = \sum_{i=1}^{n} \ln r_i$ قرار دهید. آنگاه

$$\prod_{i \in P} r_i + \prod_{i \notin P} r_i = e^{\sum_{i \in P} \ln r_i} + e^{d - \sum_{i \notin P} \ln r_i}$$

که تساوي هنگامي اتفاق مي‌افتد که $\sum_{i \in P} \ln r_i = \frac{d}{2}$ باشد.

بررسي حیاتي این است که $2e^{\frac{d}{2}}$ برابر $2\prod_{i=1}^{n} h_i$ که عدد گویایي است، مي‌باشد. پس هرگاه مسئله PSMP براي نمونه ساخته‌شده حل‌شود و جواب کمینه آن به دست آید، مي‌توان تعیین کرد P وجود دارد

که $\sum_{i \in P} \ln\ r_i = \frac{d}{2}$ باشد یا نه. معادله می‌توان به صورت $\sum_{i \in P} \ln\ h_i = \frac{1}{2}\sum_{i=1}^{n} \ln\ h_i$ نیز نوشته شود که نتیجه می‌دهد:

$$\sum_{i \in P} \ln\ h_i = \sum_{i \notin P} \ln\ h_i$$

و این در واقع نشان می‌دهد $\prod_{i \in P} h_i + \prod_{i \notin P} h_i$ که معادل $\prod_{i \in P} s_i + \prod_{i \notin P} s_i$ است. پس برای نمونه PSMP جواب وجود دارد اگر و تنها اگر برای MP جواب وجود داشته باشد.

¤

***قضیه**: مسأله ROPAS که تنها ۲ کار مستقل دارد (تعداد کارگران غیر ثابت است) سخت تمام است.*

اثبات: یک کاهش تورینگ با زمان چندجمله‌ای از PMSP به ROPAS ارائه می‌شود. r_i های دلخواه که $0 < r_1, r_2, ..., r_n < 1$ را در نظر بگیرید. گراف جهتدار بدون تنها از ۲ کار تشکیل‌شده که آنها را کار راست و کار چپ می‌نامیم. n کارگر داریم. احتمال موفقیت هر کارگر i، $p_{i,left}, p_{i,right}$ ، برابر $1 - r_i$ ، یا به‌صورت ساده با p_i نشان داده می‌شود.

انتسابی که زمان تکمیل را کمینه می‌کند را در نظر بگیرید. L مجموعه‌ای از کارگران است که به کار چپ در $\sum(\phi)$ یعنی هنگامی‌که هر دو کار مجاز هستند، منسوب می‌شوند. زمان تکمیل کمینه است اگر و تنها اگر $\prod_{i \notin L} r_i + \prod_{i \in L} r_i$ کمینه باشد. بنابراین انتساب بهینه جواب مسئله PMSP است.

براي نشان دادن معادل بودن، انتساب بهينه را بررسي مي‌كنيم. R را مجموعه‌اي از كارگران كه توسط $\sum(\phi)$ به كار راست منسوب مي‌شوند را در نظر بگيريد. L و R مجموعه‌هاي مجزا و در واقع مكمل از {1,2,....,n} هستند. $l = 1 - \prod_{i \in L}(1 - p_i)$ احتمال موفقيت حداقل يكي از كارگران منسوب‌شده به كار چپ است و $r = 1 - \prod_{i \in R}(1 - p_i)$ احتمال موفقيت حداقل يكي از كارگران منسوب‌شده به كار راست است. با استفاده از فرمول زمان تكميل را محاسبه مي‌كنيم.

$$T_{\phi} = \frac{1 + l(1-r)T_{\{left\}} + r(1-l)T_{\{right\}}}{1-(1-l)(1-r)}$$

كه $T_{\{left\}}$ زمان تكميل با شروع انتساب از كار انجام‌شده چپ و $T_{\{right\}}$ زمان تكميل با شروع انتساب از كار انجام‌شده راست است.

مي‌توان فرمول را با توجه به اينكه $T_{\{left\}}$ با $T_{\{right\}}$ برابر هستند، ساده كرد. در واقع مجموعه L و R ثابت هستند و T_{ϕ} كمينه مي‌شود هرگاه $T_{\{left\}}$ و $T_{\{right\}}$ كمينه شوند.

فرض كنيد $k \subseteq \{1,2,...,n\}$ مجموعه‌اي از كارگران باشد كه $\sum(\{right\})$ به كار چپ منسوب مي‌شوند يعني كار راست انجام‌شده و كار چپ تنها كار باقي‌مانده و مجاز است. زمان تكميل دور و انتظار در اين صورت

$$T_{\{right\}} = \frac{1}{1 - \prod_{i \in k}(1-p_i)}.$$

از آنجايي که $0 < p_i < 1$ زمان انتظار به‌صورت واضح درصورتي که $k = \{1,2,...,n\}$ باشد کمينه مي‌شود. در نتيجه $T_{\{left\}} = T_{\{right\}}$. فرض کنيم اين زمان انتظار را با T نشان دهيم. آنگاه

$$T_\phi = \frac{1 + l(1-r)T + r(1-l)T}{1-(1-l)(1-r)} = T + \frac{1 - r*l*T}{l + r - r*l}$$

توجه کنيد که $\sum(\phi)$ هر کارگر را به کار منسوب مي‌کند. براي نشان دادن اين، نشان داده مي‌شود که زمان تکميل کاهش مي‌يابد هنگامي که l يا r در آن افزايش مي‌يابد. توجه کنيد که فرمول متقارن با توجه به r يا l متقارن است پس تنها مشتق جزئي آن را نسبت به r حساب مي‌کنيم:

$$\frac{\partial}{\partial r} = \frac{-1 + l - Tl^2}{(l + r - l*r)^2}$$

اين مشتق به ازاي $0 < l < 1$ اکيدا منفي است. بنابراين زمان انتظار کمينه مي‌شود هرگاه r و l بيشينه شوند. اين معادل اين است که تمامي کارگران به کاری منسوب مي‌شوند و در واقع L و R افراز $\{1,2,...,n\}$ باشند.

پس تا به حال نتايج بدين‌صورت بوده است که انتساب در ابتدا همه کارگران را به کاري منسوب مي‌کند و هنگامي‌که تنها يک کار براي انجام‌شدن باقي مانده است انتساب کليه کارگران را به اين کار منسوب مي‌کند.

از آنجايي كه L و R افراز $\{1,2,...,n\}$ هستند، l+r-lr احتمال اينكه حداقل يک كارگر موفق به انجام كارش شوند اينست كه مستقل از اينكه كارگرها چگونه به كارهاي راست و چپ منسوب شوند تعيين مي‌شوند. در نتيجه اين احتمال كمينه مي‌شود هرگاه lr بيشينه شود.

lr در واقع:

$$lr = 1 - (\prod_{i\in L}(1-p_i) - \prod_{i\in R}(1-p_i)) + (\prod_{i\in L}(1-p_i) * \prod_{i\in R}(1-p_i))$$

كه از آنجايي كه L و R افراز $\{1,2,...,n\}$ هستند پرانتز سوم به L و R بستگي ندارد. بنابراين معادله بيشينه مي‌شود هرگاه $\prod_{i\in L} r_i + \prod_{i\in R} r_i$ بيشينه شود.

بنابراين يک انتساب بهينه انتسابي است كه $\prod_{i\in L} r_i + \prod_{i\notin L} r_i$ را بيشينه كند.

¤

برنامه ريزي گراف عريض با كارگران ثابت، سخت تمام است. هنگامي‌كه تنها ۲ كارگر و عرض گراف مي‌توان غيرثابت باشد مسئله سخت تمام است. براي رسيدن به اين هدف، ابتدا سخت تمام بودن مساله هاي كمكي نشان داده مي‌شود. در انتها يک كاهش تورينگ با زمان چندجمله‌اي نشان داده مي‌شود ROPAS با كارگران ثابت نيز سخت تمام است.

۵ توضیح کد

۵-۱ پیاده‌سازي الگوریتم براي مساله اول

مساله يك گراف كه بيانگر وابستگي كارها و n كارگر به طوري كه كارگر i با احتمال $0 \le P_{i,j} \le 1$ كار j را به درستي انجام مي دهد و براي هر كار j حداقل يك كارگر وجود دارد كه $0 < P_{i,j}$.

هدف پيدا كردن برنامه ريزي بهينه *Σ است كه «زمان تكميل» را كمينه كند.

با فرض ثابت بودن عرض گراف و تعداد كارگران مي توان در زمان چند جمله اي اين مساله را حل كرد.

الگوريتم ارايه شده به اين صورت است كه اگر ترتيب ترتيب مكاني رلس هاي گراف تكامل $Y_1, Y_2, \ldots, Y_m$ باشد. بر خلاف ترتيب از Y_m شروع مي كنيم. در هر مرحله براي Y_h كليه انتساب هاي ممكن را در نظر مي گيريم و براي آنها T_{Y_h} را با توجه به فرمول $Tx_0 = \frac{1}{a_1 + a_2 + \ldots + a_K}(1 + \sum_{i=1}^{k} a_i Tx_i)$ به دست مي آوريم. در نهايت هر انتساب $\sum(X_h)$، كه كمترين زمان تكميل را دارد به عنوان انتساب بهينه $\sum^*(X_h)$ و زمان تكميل $T^*_{x_h}$ آن زمان تكميل اين رلس هاست.

۵-۱-۲ مسايل مطرح در پياده سازي

در پياده سازي اين الگوريتم مسايل زير مطرح است :

۱) تعريف ثابت هاي مورد استفاده در برنامه

a) بيشترين تعداد کارگران

```
const int MAX_N = 10
```

b) بيشترين تعداد کارهايي که ممکن است برنامه با آن تست شود.

```
const int MAX_T = 10
```

۲) متغير هاي عمومي

a) ورودي هاي مساله

i) تعداد کارها (t)، تعداد کارگران (n) و احتمال انجام هر کار توسط هر کارگر (ماتريس دو بعدي p).

ii) درجه ورودي و خروجي هر راس در گراف پيش نيازي کارها

```
int ind[MAX_T], outd[MAX_T]
```

iii) ليست فرزندان (راس خروجي) و پدران (راس هاي ورودي) هر راس

```
vector <int> inM[MAX_T],outM[MAX_T]
```

b) براي اثبات اين كه زمان تكميل بهينه متناهي است يک حد براي اين زمان ارايه شد. اين مقدار را در متغير limit ذخيره مي كنيم.

c) در هر مرحله اولين عنصر صف را براي به دست آوردن انتساب بهينه ي آن و زمان تكميل بهينه ي آن پردازش مي كنيم و آن را از اول صف برمي داريم و درصورتي كه كليه پدران اين راس را كه در صف نيستند به انتهاي صف اضافه مي كنيم هر عنصر اين صف از نوع DAG است.

```
vector <DAG> tdags
```

d) يک نگاشت[1] از X (براي هر DAG، X آن شناسه ي آن است.) ها به زمان تكميل بهينه ي آن X براي استفاده در محاسبه T_{x_h} (x_h پدران X). در ابتدا كه يک X به انتهاي صف اضافه مي شود به اين نگاشت X با زمان limit را اضافه مي كنيم.

```
map < bitset<MAX_T> , double, eqbits> m
```

۳) تعريف كلاس DAG كه راس هاي آن در ۳ گروه X، E(X) وساير راس ها تقسيم شده اند. چند متغير كمكي ديگر كه براي سهولت در محاسبات و در واقع بهتر كردن زمان اجراي الگوريتم.

```
class dag{
```

[1] Map

```
        bitset<MAX_T> x;

vector<int> eX;



vector<int> sinks; // local sinks(considering x as whole dag)

int outdeg[MAX_T];// out degree each node of x( considering x as whole dag) when outdeg one node equals to 0 it is local sink, when it is equal to outd of this node means all its children are in x now. Else some of its children are in E(x).

double prob[MAX_T]; // when sigma is determined , meanwhile for each task ti  in sigma prob[ti] equals to probability of failure of all workers

        int sig[MAX_N]; //storing sigma

   };
```

۴) توابع کمکي

a) تابع محاسبه پدران يک رلس در گراف تکامل براي اينکه آن ها را درصورتي که درصف نيستند به آن اضافه کنيم.

```
void gen(dag d){

   bitset<MAX_T> subs(0ul);
```

```
int i ;
int w = d.sinks.size();
do{
	dag sol(d.x, d.sinks , d.eX );
	for (int j =0 ; j < MAX_T; j++)
		sol.outdeg[j]= d.outdeg[j];

	//producing all subsets
	i= 0;
	while (i <w  && subs[i]){
		i++;
	}
	if (i < w ){
		subs[i] = 1;
		for (int j =0 ; j < i ; j++)
		subs[j]=0;
		// subset is determined now.
```

```
vector<int> news;
for (int j = w-1; j >=0 ; j--)
if (subs[j]){
int v = sol.sinks[j];
bitset<MAX_T> childs(0ul);
for (int k =0 ; k < outd[v] ; k++)
childs[outM[v][k]] = 1;
for (int k = sol.eX.size()-1; k >= 0 ; k--)
if(childs[sol.eX[k]])
sol.eX.erase( k);
sol.eX.push_back(v);
sol.x[v] = 0;
}
if ( m.find(sol.x) == m.end() ){
for (int j = w-1; j >=0 ; j--){
int v = sol.sinks[j];
```

```
if (subs[j]){
for (int k = 0 ; k < ind[v]; k++){
int parv = inM[v][k];
sol.outdeg[parv]++;
if(sol.outdeg[parv]== outd[parv])
news.push_back(parv);
}
sol.sinks.erase(j);
}
}
for (int j = 0; j < news.size(); j++)
sol.sinks.push_back(news[j]);
for (int j = 0; j < sol.eX.size(); j++)
sol.prob[sol.eX[j]] = 1;
tdags.push_back(sol);
m[sol.x] = limit;
}
```

```
        }
    }while(i < w );// produce 2 ^ |x| subsets (|x| < w)
}
```

b) تابع محاسبه کليه انساب هاي ممکن براي يک DAG.

```
void allSch(int ind){
    if (ind == n ){
        findT();
        return;
    }
    for (int i =0 ; i < tdags[0].eX.size(); i++){
        int v = tdags[0].eX[i];
        sigma[ind] = v;
        bool notin = true;
        for (int j =0 ; j < scht.size(); j++)
            if (scht[j] == v)
            notin = false;
```

```
            if ( notin )
                scht.push_back(v);
            tdags[0].prob[v] *= (1-p[ind][v]);
            allSch(ind+1);
            tdags[0].prob[v] /= (1-p[ind][v]);
            if ( notin )
                scht.pop_back();
    }
}
```

c) تابع محاسبه ي زمان تكميل براي انتساب scht به ازاي عضو اول صف.

```
void findT(){
    int w = scht.size() ;
    bitset<MAX_T> subs(0ul);
    double time = 1;
    int i;
    // producing all subsets
do{
```

```
i= 0;
while (i <w  && subs[i]){
        i++;
}
if (i < w ){
        subs[i] = 1;
        for (int j =0 ; j < i ; j++)
        subs[j]=0;
        //subset is now produced
        bitset<MAX_T> next = tdags[0].x;
        double p = 1;
        for (int j =0 ; j < w; j++){
        int v = scht[j];
        if (subs[j]){
        p *= (1-tdags[0].prob[v]);
        next[v] = 1;
        }else
```

```
                p *= tdags[0].prob[v];
                }
                time += p * m[next];
        }
}while(i < w); );// produce 2 ^ |x| subsets (|x| < w)
double allfail=1;
for (int j=0; j < w; j++){
        int v = scht[j];
        allfail *= tdags[0].prob[v];
}
time = time / (1 - allfail);


if (time < m[tdags[0].x]){
        m[tdags[0].x] = time;
        for (int j =0 ; j < n ; j++)
                tdags[0].sig[j] = sigma[j];
}
```

}

۵-۱-۳شبه برنامه ي كلي

شبه برنامه ي اصلي به اين صورت است :

```
int main(){
	produce first member of tdags which x = {1,2,...,n} and its completion time is 0.
	readInput();
do{
		gen(tdags[0]);
		if ( tdags[0].eX.size() )
			allSch(0);
		tdags.erase(tdags.begin());
	}while (tdags.size());
}
```

۵-۱-۴ تحليل زمان اجراي پياده سازي

به ترتيب شبه برنامه تحليل مي كنيم (فرض مي كنيم تعداد كل راس هاي گراف تكامل V است) :

۱. تولید عنصر اول در $O(t)$.

۲. خواندن ورودي از دو قسمت تشكيل‌شده است :

a. گراف وابستگي كارها در $O(E)$، كه E تعداد يالهاي اين گراف كه $O(t^2)$ است.

b. احتمال انجام كارها توسط كارها و تعداد كارگران در $O(nt+n)$.

۳. زمان اجراي تابع gen ، $O((2^w+w)*V)$ كه با توجه به ثابت بودن w ثابت است.

۴. زمان اجراي تابع allSch : براي هر يكي از $(w+1)^n$ انتساب ممكن زمان اجراي آن را در $O((2^w+w)*V+w)$ محاسبه مي كند و در كل مي‌شود $O((w+1)^n*((2^w+w)*V+w))$.

و در كل زمان براي پردازش داريم، پس :

$$O(t+t^2+nt+n+V*(((2^w+w)*V)+(w+1)^n*(((2^w+w)*V)+w)))$$

و از آنجايي كه با توجه به توضيحات V،$O((t+1)^w)$ است به صورت ساده شده مي توان زمان اجراي الگوريتم:

$$O(V*(t+t^2+(C_1*V)+(C_2*V)))$$

كه بر حسب t چندجمله اي است.

۵-۲ پیاده‌سازي الگوریتم کارگران مارکوفی

مساله یك گراف كه بیانگر وابستگي كارها و n كارگر به‌طوري كارگر i با احتمال $0 \le P_{i,j} \le 1$. كار j را به درستي انجام مي دهد و براي هر كارگر یك $0 < P_{ij}$. به علاوه هر كارگر یك DTMC، M_i دارد كه متشكل از حالت هایي و انتقال بین آنها است و هر حالت s_k، یك $0 < q_{sk} < 1$ دارد كه احتمال انتقال توسط موفقیت را در این حالت نشان مي دهد و بدین ترتیب احتمال موفقیت در انجام یک كار براي یک كارگر ثابت نیست و برابر $q_{s_k} * P_{i,j}$ است.

هدف پیدا كردن برنامه ریزي بهینه Σ^* كه زمان تكمیل را كمینه كند.

با فرض ثابت بودن طول دورها در گراف شكست عمومي و عرض گراف و تعداد كارگران مي توان در زمان چند جمله اي این مساله را حل كرد.

۵-۲-۱ مسایل مطرح

تنها تفاوت كه مساله در این حالت با مساله اول دارد این است كه در این جا به ازاي هر رلس در گراف تکامل، یک گراف شكست عمومي داریم كه بدون دور هم نیست. الگوریتم به این‌صورت یود كه همانند مساله اول بر خلاف ترتیب ترتیب مكاني رلس هاي گراف تکامل حركت مي كنیم. سپس در هر رلس ابتداسعي مي كنیم انتساب بهینه و زمان تكمیل آن را به ازاي حالت هاي مختلف كارگران به هست آوریم. براي این كار به ازاي هر انتساب ممكن، زمان تكمیل :

۱. براي رئوس هاي روي دورها. براي اين كار بايد كليه انتساب هاي ممكن براي اين رئوس ها را در نظر بگيريم و به ازاي $((w+1)^n)^r$ معادله ي به دست آمده زمان تكميل بهينه براي رئوس اول را به دست مي آوريم.

۲. رئوسهاي مسير هاي گراف شكست عمومي به ترتيب فاصله آنها از دورها پردازش مي كنيم و زمان تكميل هر انتساب را در $O(2^n)$ با توجه به سناريو هاي مختلف براي عملكرد كارگران به دست مي آوريم.

۵-۲-۲ تحليل زمان اجراي پياده سازي

الگوريتم پياده سازي شده براي گراف شكست عمومي است كه طول دور ها در آن ۱ باشد. در اين برنامه ابتدا سعي مي شود ترتيب پردازش رئوس هاي گراف شكست عمومي مشخص شود. براي اين كار ابتدا رئوس هاي داراي طوقه (nself) وسپس ساير رئوس ها (nother) به ترتيب فاصله آنها از دور مي آيند.

در اينجا تفاوتي كه وجود دارد در نحوه ي محاسبه ي احتمال انتقال كه در $O(2^n)$ به جاي $O(2^w)$ است. تحليل زماني الگوريتم به ترتيب اجراي آن به اين صورت است (با فرض اينكه $|M_i|$، M و تعداد رئوس هاي گراف تكامل V باشد) :

1. توليد عنصر اول در $O(t^*M^n)$.
2. خواندن ورودي از دو قسمت تشكيل شده است :
   a. گراف وابستگي كارها در $O(E)$، كه E تعداد يالهاي اين گراف كه $O(t^2)$ است.
   b. احتمال انجام كارها توسط كارها و تعداد كارگران در $O(nt+n)$.

c. گراف مارکوف کارگران در $O(2M*n)$.

۱. زمان اجراي تابع gen ، $O((2^w+w)*V)$ كه با توجه به ثابت بودن w ثابت است.

۲. زمان اجراي تابع allSch : براي هر يكي از $(w+1)^n$ انتساب ممكن زمان اجراي براي حالت هر حالت ممكن كارگران در $O(2^n*V)$ محاسبه مي كند.

و در كل رلس براي پردازش داريم، پس :

$$O(t*M^n+t^2+nt+n+2M*n+V*(((2^w+w)*V)+(w+1)^n*((2^n)*V)))$$

و از آنجايي كه با توجه به توضيحات V،$O((t+1)^w)$ است به صورت ساده شده مي توان زمان اجراي الگوريتم:

$$O(V*(t+t^2+(C_1*V)+(C_2*V)))$$

كه بر حسب t چندجمله اي است.

پیوست ۱ : واژگان انگلیسی به فارسی

Admissible evolution............تکامل مجاز

Arc............کمان

Backward............حرکت رو به عقب

Challenge............دغدغه

Child............فرزند

Distinct subsets............زیر مجموعه هاي مجزاي

Eligible............مجاز

Expected completion time............زمان تکمیل مورد انتظار

Fail............شکست

Grid computing............محاسبات شبکه اي

Heuristic approach............روش حریصانه

Independent............مستقل

Induced graph............گراف القاشده

Internal node............رلس هاي داخلي

Large-scale............مقیلس بزرگ

Optimal assignment............انتساب بهینه ي

Overlap...اشتراك

Parallel scheduling..برنامه ريزي موازي

Parent..پدر

Path...مسير

Precedence Constraint...پيش نيازي

Project management...مديريت پروژه

Sink...چاه

Sophisticated..پيچيده

Source..چشمه

State..حالت

Success..موفقيت

Stochastic ....تصادفي

Transfer...انتقال

Uncertainty..عدم قطعيت

Unreliable...غير قابل اعتماد

پیوست ۲ : واژگان فارسی به انگلیسی

انتساب بهینه ي..Optimal assignment

انتقال..Transfer

برچسب گذاري...Label

برنامه ريزي موازي...Parallel scheduling

پدر..Parent

پيچيده..Sophisticated

ترتيب مكاني..Topological sort

تصادفي... Stochastic

دغدغه...Challenge

زمان تكميل ....Completion time

زنجيره ماركوف زمان گسسته...Discrete time markov chain

زير مجموعه هاي مجزاي...Distinct subsets

State...حالت

Backward...حرکت رو به عقب

Sink...چاه

Source..چشمه

Cycle..دور

Internal node..رئوس هاي داخلي

Heuristic approach...روش حريصانه

Expected completion time..زمان تکميل مورد انتظار

Precedence Constraint...پيش نيازي

Fail..شکست

Uncertainty..عدم قطعيت

Unreliable...غير قابل اعتماد

Child...فرزند

Arc .... کمان

Induced graph .... گراف القاشده

Directed Acyclic Graph(DAG) .... گراف جهت دار بدون دور

Project management .... مدیریت پروژه

Grid computing .... محاسبات شبکه ای

Eligible .... مجاز

Independent .... مستقل

Path .... مسیر

Large-scale .... مقیاس بزرگ

Success .... موفقیت

فهرست منابع

[1] G. Malewics. Parallel scheduling of complex dags under uncertainty. In SPAA'05: Proceedings of the seventeenth annual ACM symposium on Parallelism in algorithms and architectures, pages 66-75, New York, NY, USA, 2005,ACM.

[2] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, Introduction to Algorithms. McGraw-Hill Higher Education, 2001.

[3] R. P. Dilworth. A decomposition theorem for partially ordered sets. Annals of Mathematics, 51: 161-168, 1950.

[4] J. Annis, Y.Zhao, J. Voeckler, M. Wilde, and I. Foster. Applying chimera virtual data concepts to cluster finding in the sloan sly survey. 5th conference on High Perfomance Networking and computing(SC), 56-69, 2002.

[5] M. Narasimhan , and J. Ramanujam. A fast approach to computing exact solutions to the resource-constrained scheduling problem. ACM Transactions on Design Automation Electronic Systems, Vol. 6(4): 490-500, 2001.

[6] M. Mori and C. Tseng. A resource constrained project scheduling problem with reattempt at failure: A heuristic approach. Journal of the operation research society of Japan, Vol. 40 : 33-44, 1997.

[7] C.C. Tseng, M.Mori, and Y. Yajima. A project scheduling model considering the success probability. Proc. Of the association of Asian Pacific Operational Research Socities(ARPOS),399-406, 1994.

[8] W. Herroelen, and P. Leus. Project scheduling under uncertainty, Survey and research potential. European Journal of Oprational Research, Vol. 24 : 332-337, 1977.

[9] A. Fernandez, R. Armcost, and J. Pet-Edward. Understanding simulation solutions to Resource constrained project scheduling problems with stochastic task duration. Engineering management Journal, Vol. 10: 5-13, 1998.

[10] A. Fernandez, R. Armcost, and J. Pet-Edward. A model for the Resource constrained project scheduling problems with stochastic task duration. 7th Industrial Engineering Research Conferance, 1998.

[11] M.A. Turnquist, and L.K. Nozick. Allocating time and resource in project management under uncertainty. 36th Annual Hawaii International Conferance on System Sciences(HICSS),2003.

[12] J. Kleinberg, Y. Rabani, and Y. Tardos. Allocating bandwidth for bursty connections. SIAM Journal on Computing. Vol. 8: 191-217, 2000.

[13] A. Goel, and P. Indyk. Stochastic load balancing and related problems. 40th Annual Symposium on Foundations of Computer Science(FOCS), 579-586, 1999.

[14] M. Skutella, and M. Uetz. Scheduling precedence constrained jobs with stochastic processing times on parallel machines. 12th ACM-SIAM Symposium on Discrete Algorithms(SODA), 589-590, 2001.

[15] D. S. Johnson: The NP-completeness column: An ongoing guide: Vol. 13@ 438-448,1987.

[16] R.Peeters, The maximum edge biclique problem is NP-complete. Discrete applied mathematics, Vol. 131: 651-654, 2003.

Scheduling of Dags under Uncertainty

by

Sara Ahmadian

Submitted in Partial Fulfillment of the Requirements for the Degree of

Bachelor of Science

in

Computer Engineering (Software)

Under supervision of

Dr. Mohammad Ghodsi

August 2008

Computer Engineering Department

Sharif University of Technology

Tehran