مشکل زمانبندی وظایف در محاسبات ابری

در رایانش ابری، خط مشی زمانبندی وظایف به طور مستقیم بر کارایی وظایف کاربر و استفاده کارآمد از منابع تحت محیط ابری

تأثیر می گذارد از این رو، چگونگی دستیابی به تخصیص بهینه وظایف کاربر مسئله کلیدی زمانبندی کارهادر رایانش ابری

است . روند کار زمانبندی در محیط ابر به شرح زیر است.کارها و منابع با توجه به وظیفه فعلی و اطلاعات منابع مطابق با استراتژی خاص نقشه برداری می شود سپس نقشه بین منابع اختصاص داده شده دراجرای کار برای اطمینان از کارایی

وظیفه و کیفیت خدمات مورد نیاز کاربران دنبال می شود . سرانجام خلاصه نتایج برای کاربر ارسال کننده اجرا می شود.

محیط محاسبات ابری فعلی بیشتر با توجه به مدل برنامه نویسی MapReduce که یکپ مدل زمانبندی کار کارآمد به ویژه برای نسل و پردازش مجموعه داده های بزرگ می باشد ساخته شده است [۲۰-۱۶]

فرآیند پیاده سازی در شکل ۱ نشان داده شده است .شکل ۱ مدل پیشنهادی برای زمانبندی کارها در محیط ابر را که شامل دو مرحله نگاشت و کاهش می باشد تشکیل شده است . ایده اصلی نگاشت کاهش پردازش موازس کارها به دو مرحله نگاشت و کاهش می باشد .در مرحله نگاشت کارهای کاربران به تعدادی زیر کار بوسیله تابع نگاشت کاهش تقسیم شده و به چندین پردازشگر (کارگر) تخصیص داده می شوند سپس یک فایل میانی خروجی وجود خواهد داست . در مرحله کاهش نتایج پس از پردازش تحلیل نگاشت یک پارچه شده از پیش نتیجه به خروخی داده می شود .

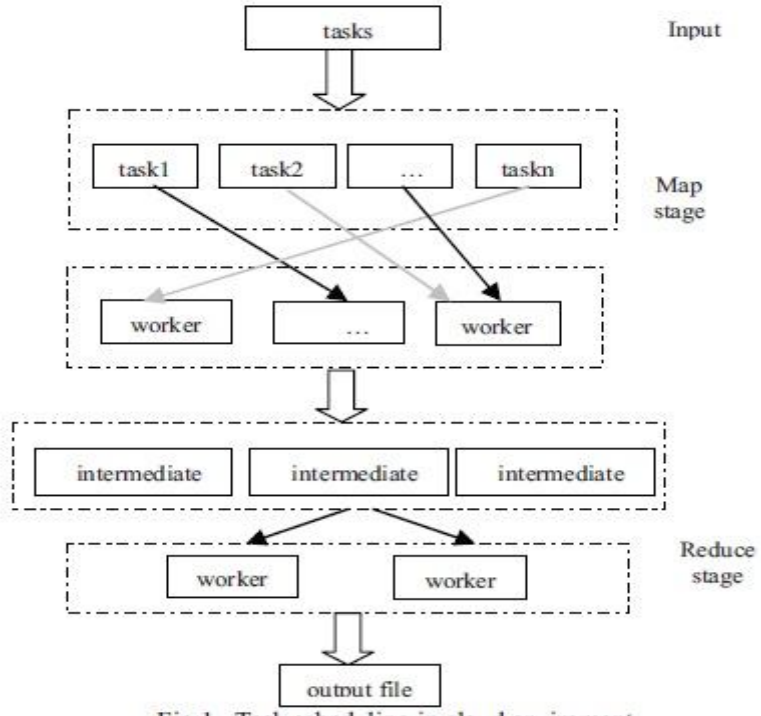

Fig 1 Task scheduling in cloud enviroment

الگوریتم زمانبندی کار بر اساس الگوریتم ژنتیک کلونی مورچه ها در محاسبات ابری

هدف این مطالعه بررسی مسئله زمانبندی کارها در پردازش ابری است . به منظور بدست آوردن بهترین نتیجه از زمان بندی کار

در کمترین زمان ممکن الگوریتم ژنتیک و کلونی مورچی ها یک پارچه شده که می تواند استناد شود به مدل نگاشت کاهش . بر این اساس، یک الگوریتم GA-ACO پیشنهاد می شود

ایده های طراحی

ایده اصلی الگوریتم GA-ACO به شرح زیر است. در مراحل اولیه زمانبندی کار، از مزیت جستجوی سراسری الگوریتم ژنتیک و کد گذاری غیر مستقیم کروموزن ها استفاده م شود سپس انعکاس زمان اتمام کار به عنوان تابع قابلیت انتخاب می شود . پس از انتخاب جهش و تقاطع راه حل بهینه را می سازد و این راه حل را به فرمون اوئلیه aco تبدیل می کند . و راه حل بهینه زمانبندی کار را از طریق ویژگی بازخورد مثبت و کارایی تشکیل می شود

تشکیل جمعیت اولیه:

در مرحله اول یک جمعیت اولیه تولید میشود که شامل تعدادي کروموزوم هست. هر کروموزوم شامل تعدادي ژن هست. در طرح

ما، جابها ژنهاي طرح هستند كه ماشينهاي مجازي به اين جابها تخصيص داده ميشوند. مجموعه ژنهاي ما تشكيل كروموزوم را ميدهد.

جابها بايد به اين نودها تخصيص داده بشوند و ميتوان فرض كرد ايندكس هاي خانه هاي آرايه هستند. يعنی ايندكس ژنها)نودها(هستند.

در مرحله توليد جمعيت اوليه، تعدادي كروموزوم ساخته ميشوند كه طولشون به تعداد جابها است و تعدادشون متغير هست و به صورت

رندوم اين جابها به ماشينهاي مجازي تخصيص داده ميشوند. فرض كنيد كه چهار تا جاب از شماره يک تا چهار در حال حاضر داريم. ما

تعدادي كروموزوم ميسازيم كه ژنهاي آن همون چهار تا جاب هستند. ولی در هر كروموزوم اين جابها به vmهاي خاصی تخصيص داده

شده.)اين تخصيص دادن رندوم است(. مثلا در كروموزوم اول جاب اول به vm شماره سه و در كروموزوم دوم همين جاب)جاب اول يا

همون ژن اول)مثلا به vm شماره شش و در كروموزوم سوم همين جاب(جاب اول(مثلا به ماشين مجازي شماره يک تخصيص داده

شده. همين روال براي ژنهاي شماره دو تا چهار برقرار است. و هر كدوم از اين ژنها به vm خاصی تخصيص داده ميشوند. تعدا كروموزومها

هم گفتيم كه متغير هست. در بقيه كروموزومها همين روال هست. يعنی هر ژن كروموزوم كه بيانگر جاب خاصی هست)در اين مثال چهار

ژن وجود داره(در هر كروموزوم به vm خاصی سپرده ميشه.)پايان مرحله اول يا همان جميعت اوليه(

قوانين الگوريتم ژنتيک

رمزگذاری و رمزگشايی كروموزوم برای حل مشكل زمان‌بندی كارها تحت محيط ابر . ما بايد طرح زمانبندی كارها را در كروموزن ها كد كذاری كنيم كه هر كروموزون طرح كدگذاری خاصی را ارائه می كند . اين مقاله از يک روش رمزگذاری غير مستقيم استفاده می كند.

عمليات خاص به شرح زير است: هر كاری يک منبع كد گذاری شده را اشغال می كند . طول كروموزوم برابر است با تعداد كل زير كارها تعداد هر موقعيت بيت نشان دهنده تعداد زير وظايف ژن و مقدار ژن – بيت نشان دهنده تعداد منبع اشغال شده است

مقدار زيركارها (كارهای فرعی) با استفاده از محاسبه می شود

$$\text{subtasknum} = \sum_{t=1}^{m} task\ number\ (t)$$

m = number of tasks
t = the order of task
taskNum(t) = the number of sub-tasks assigned to task t
For example, it assumes that there are 3 tasks ,then *m*=3.And 3 resources means *n*=3 and 3 tasks are divided into 3,4,2 sub-tasks, meaning that *taskNum(1)*=3, *taskNum(2)*=4, *taskNum(3)*=2,so *subtaskNum*=9.It means that the length of the chromosome is 9.

تنظیم محدوده ارزش ژن به (۱، ۳). باشد از روش رمزگذاری غیر مستقیم برای تولید

مجموعه ای از کروموزوم ها : {۲، ۳، ۱، ۲، ۳، ۱، ۲، ۱، ۱}. استفاده می کند کار فرعی اول به منبع دوم، کار فرعی دوم به منبع سوم و غیره. سپس این کروموزوم ها رمزگشایی می شوند و منابع مختلف توزیع شده روی کارها

اولین کار فرعی به منبع دوم، کار فرعی دوم به مبع سوم و غیره اختصاص داده می شود . سپس این کروموزوم ها رمزگشایی شده و توزیع منابع مختلف بر روی کارها مشخص می گردد . w1:{3,6,8,9} w2:{1,4,7} w3:{2,5}

۲- تابع هدف و تابع برازندگی زمان اجرا ی انجام کارها برای هر منبع را با استفاده از توالی رمزگشایی شده و ماتریس ETC محاسبه می کند.زمان بندی منبع برای تکمیل کار به شرح زیر است :

$$f(x) = max \sum_{i=1}^{w} work(\text{r و } i)$$

در حالی که :

Work(r, i) : زمان صرف شده توسط منبع r برای انجام زیرکار i روی هر منبع

W: میزان زیر کارهایی که به منبع اختصاص داده شده

معادله بالا به عنوان تابع هدف تعریف شده است . تابع برازندگی برای ارزیابی مزایا و معایب کروموزوم ها استفاده می شود مقدار تابع بزرگتر است، سپس بقای کروموزوم ها قوی تر و راه حل تابع بهتر است و زمان کوتاهتر و ارزش سازگاری بیشتر و احتمال انتخاب شدن بیشتر است.

تابع سازگاری به صورت زیر تعریف می شود : f(x) = 1/F(x)

۳- دستکاری ژنتیکی : دستکاری ژنتیکی الگوریتم ژنتیک شامل انتخاب، ترکیب و جهش می باشد و از طریق این عملیات به تولید افراد جدید ادامه می دهد تا راه حل بهینه را جستجو کنند.

- انتخاب : احتمال انتخاب برای هر فرد بر اساس مقدار تابع برازندگی. محاسبه می شود معادله زیرنحوه محاسبه احتمال انتخاب را نشان می دهد:

$$p(i)= \frac{f(i)}{\sum_{j=1}^{n} f(j)!}$$

ترکیب :

این مقاله روش‌های ترکیبی تطبیقی را انتخاب می‌کند . ترکیب‌بزرگ احتمالاً تعدادی از بیت ها را بین افراد تغییر می دهد بنابراین می تواند از رخداد زود رس جلوگیری کند . در بخش پایانی الگوریتم احتمال ترکیب کاهش می یابد و این آسانتر است برای ایجاد یک نسل خوب و تسریع سرعت همگرایی

جهش :

این مقاله از جهش تک نقطه ای برای تغییر برخی بیت های انفرادی در گروه ها برای احتمال کمتر، مانند "۱" به

"۰"و"۰" به "۱". استفاده می کند . در عملیات واقعی، افراد جدید را که مقدار تابع سازگاری کمتر از مقدار متوسط دارند را

پس از چندین تکرار بازگشتی حذف می کند و راه حل بهینه از گروه هایی معین به عنوان مبنایی برای بدست آوردن فرمون کلونی مورچه ها بدست می آید .

راه حل دقیق براساس کلونی مورچه ها :

۱- ترکیب الگوریتم ژنتیک و کلونی مورچه ها جمعیت کروموزوم را مطابق نرخ تکامل ٥ نسل متوالی که کوچک است ارزیابی می کند . سپس الگوریتم ژنتیک می تواند خاتمه یابد و الگوریتم کلونی مورچه ها اجرا شود .

وقتی الگوریتم ژنتیک خاتمه یافت، افراد در جمعیت با توجه به مقدار تابع برازندگی مرتب می شوند که از میان آن ۱۰ درصد افراد به عنوان راه حل بهینه انتخاب و تبدیل به فرمون اولیه می شوند

نقش ارزش دهی اولیه معین در زیر نشان داده شده است

$$T_i^g = ps_n$$

p : ثابت خود تنظیم

S_n: به عنوان راه حل بهینه الگوریتم ژنتیک از طریق عملکرد الگوریتم ژنتیک در توزیع فرمون می توانیم به دست آوریم مقدار اولیه فرمون منبه در رابطه زیر مرتب می شود .

$T_i(0)=r_i+t_i^g(0)$ در حالی که :

R_i: ظرفیت پردازش منبع

T_i^g: مقدار فرمون تبدیل شده از راه حل بهینه زمانی که الگوریتم ژنتیک فعلی خاتمه یافته است

۲- انتخاب مسیر

هر مورچه با توجه به اطلاعات منبع فعلی، احتمال منبع بعدی را تعیین می کند.

$$P_{k,}(i,j)=\frac{[T_j(t)]^{\alpha}[\eta_j]^{\beta}}{\sum_{\in U}([T_u(t)]^{\alpha}[\eta_u]^{\beta})}$$

T_j (t) : مقدار فرمون در منبع j درلحظه t

η_j: ظرفیت پردازش منبع j

α or β : اهمیت فرمون

۳- به روز رسانی فرمون

با مقایسه عملکرد ACO، روش به روز رسانی سراسری فرمون می تواند همگرایی را بهبود بخشد یعنی وقتی مورچه ای انتخاب منبع را با موفقیت کامل می کند، فرمون تغییر خواهد کرد. قانون به روز رسانی فرمون در زیر نشان داده شده است

$T_j^{new}=pt_j^{old}+\Delta t_j$

در حالی که :

P: شرایط خاتمه ACO

وقتی سیکل شمارنده N از محدوده تکرار به حداکثر عدد رسید، مقدار فعلی طرح زمانبندی بهینه است و سپس ACO خاتمه می یابد

نتایج و بحث

به منظور بررسی امکان سنجی و اثربخشی الگوریتم GAACO باید آن را از روی عملکرد طزمانبندی کارها شبیه سازی کنیم . پس از آن، نتایج تجربی مقایسه می شود با الگوریتم ژنتیک و ACO تحت محیط. یکسان

- تنظیمات پارامتر

برای مزایای هر جستجو و راه حل لگوریتم ژنتیک و ACO با آزمایش های مکرر تعداد مختلفی از سناریوها با مقادیر پارامترهای مختلف در طول شبیه سازی درنظر گرفته می شود . جدول ۱ پارامترهای شبیه سازی مورد استفاده در آزمایش را بطور خلاصه نشان می دهد

نتایج تجربی و تجزیه و تحلیل

تمامی الگوریتم ها بر روی پلتفرم شبیه ساز ابر IM پیاده سازی و تست شده در آزمایش‌های انجام‌شده در این مطالعه، شکل ۲ نتایج را برای ۵۰ کار در شرایط محیطی یکسان نشان می‌دهد و مقایسه می کند میزان موفقیت در جستجوی راه حل بهینه و تعداد تکرارها در بین الگوریتم های GA-ACOو

Algorithm	Parameter	Value
GA	Number of population	100
	Crossover rate	0.6
	Mutation rate	0.1
ACO	Number of ants	100
	α	۱
	β	۱
	ρ	0.3
	a	1
	b	۰/۸

GA وACO

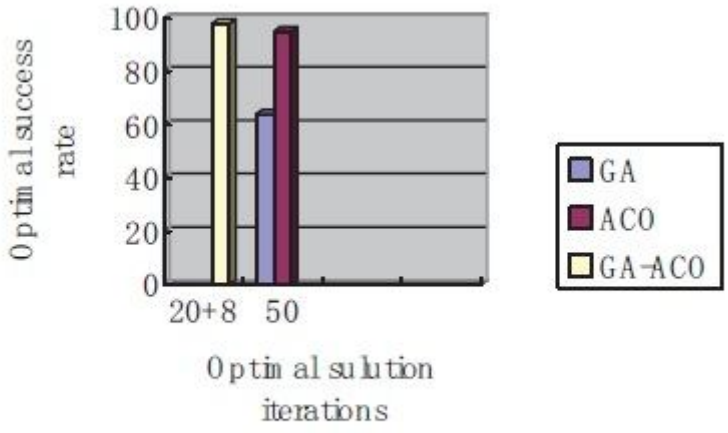

Fig.2 Comparison of optimal success rate

از شکل ۲، می توان اشاره کرد که میزان موفقیت جستجوی راه حل بهینه تا ۹۸٪ می رسد زمانی که

تعداد تکرار در آزمایش الگوریتم GA-ACOبرابر با ۲۸ بار می باشد اما زمانی که تعداد تکار الگوریتم GA ۵۰ می باشد .میزان موفقیت الگوریتم GA برابر با ۶۳% و میزان موفقیت الگوریتم ACO برابر با ۹۵% می باشد

از شکل ۲، همچنین می توان نتیجه گرفت که الگوریتم GA-ACO برای یافتن راه حل بهینه به تکرارهای کمتری نیاز دارد. و به طور قابل ملاحظه ای کارایی حل الگوریتم GA-ACO آ بهتر از GA و ACO است این به این دلیل است که الگوریتم GA-ACO چندین راه حل بهینه سازی تولید شده توسط الگوریتم GA را به فرمون ACO، تبدیل می کند و زمان جمع آوری فرمون را بسیار کوتاه می کند در آزمایش دیگری ما از الگوریتم CA-ACO، GA و ACO به ترتیب برای آزمایش زمانبندی کارها استفاده کردیم از داده های نمونه برداری شده با تعداد کار ۴۰۰’۳۰۰’۲۰۰’۱۰۰’۵۰و ۵۰۰

۵۰۰’۴۰۰’۳۰۰’۲۰۰. نمودار اثر شبیه سازی زمان اجرای کار توسط سه الگوریتم به شرح زیر است.

همانطور که از شکل ۳ می بینیم، این مشاهده زمان اجرا با افزایش تعداد کارها برای سه الگوریتم را نشان می دهد . از شکل مشخص است که وقتی تعداد کارها کمتر باشد، منابع کافی تر هستند در محیط ابری در مورد زمان اجرای کار، همه این سه الگوریتم هزینه نسبتا کمی دارند و در بین این سه الگوریتم، الگوریتم GA-ACO کمی بهتر از ACO و GA است، اگرچه شکاف آشکار نیست. با افزایش تعداد وظایف، روند افزایشی زمان اجرای صرف شده توسط CA-ACO به طور قابل توجهی کمتر از دو الگوریتم دیگر است علاوه بر این، بهبود عملکرد آشکار است دلیل اصلی این امر این است که افزایش تعداد کارها منجر به بار زیاد برای هر الگوریتم می شود که منجر به افزایش زمان اجرا می شود.

با این حال، در تعداد بیشتری از کارها، الگوریتم GA-ACO از مزایای خاص خود استفاده می کند و از نقص جستجوی محلی GA و کمبود فرومون اولیه ACO جلوگیری می کند.

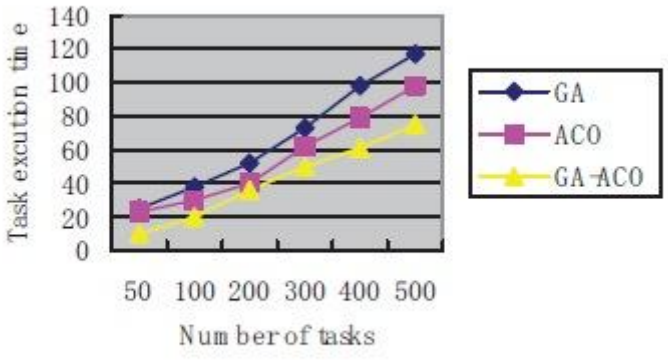

Fig 3 Comparison of task execution time

نتیجه گیری

این مقاله تحقیقاتی را در زمینه زمان بندی کارها تحت محیط ابری، با هدف حل مشکل همگرایی ناشی از فقدان اولیه

فرمون ACO انجام می دهد. سپس GA-ACO(ادغام الگوریتم ژنتیک و ACO)را معرفی می کند که از

قابلیت جستجوی سراسری قوی GA برای دستیابی به راه حل بهتر، استفاده می کند و سپس این راه حل را به فرمون اولیه ACO تبدیل می کند و در نهایت از طریق بازخورد مثبت از زمانبندی بهینه.ACO

برخوردار می شود بر اساس نتایج شبیه سازی ادغام GA و ACO برای استفاده در محاسبات ابر برای حل زمان‌بندی کارها مفید است زیرا به طور موثر کارایی جستجوی الگوریتم را بهبود می بخشد.