
بسم الله الرحمن الرحيم

دانشگاه آزاد اسلامی واحد خرم اباد

جزوه پردازش موازی

استاد:

خانم دکتر جودکی

نویسنده :

سجاد نعمتی پویا

زمستان۹۵

جلسه اول پردازش موازی

پردازش موازی:پردازش موازی یا محاسبه موازی به اجرای همزمان یک برنامه که به بخش های کوچک تری تقسیم شد بر روی چند پردازنده اطلاق می شود.

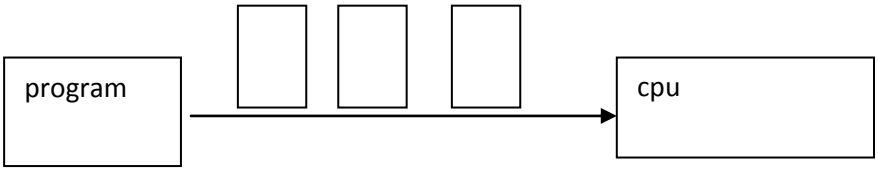
در واقع یک مسله به زیر وظیفه ای تقسیم میشود که با اجرای همزمان آن مسله اصلی در زمان کوتاه تری حل می شود.

بسیاری از برنامه ها احتیاج به پردازش حجم زیادی از داده ها دارند از جمله پایگاه های عظیم داده موتورهای جستجو وب،تصویر برداری، تشخیص پزشکی ، مدلسازی های مالی و اقتصادی ، گرافیک پیشرفته، فناوری های چند رسانه ای، شبکه های ویدئویی،

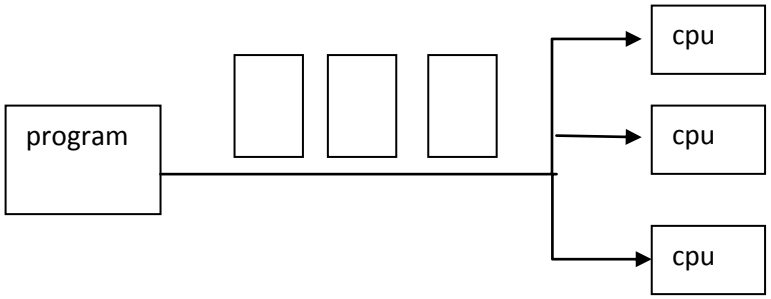
پردازش ها

پردازش سریالی

time sharing



پردازش موزی



طبقه بندی (Flynn)

ناظر بر کامپیوتر های حاوی یک یا چند پردازنده است و آنها را بر اساس نحوی تعامل بین دستور وداده متمایز میکند.

وبه چهار دسته ی :

SISD single instraction singele data

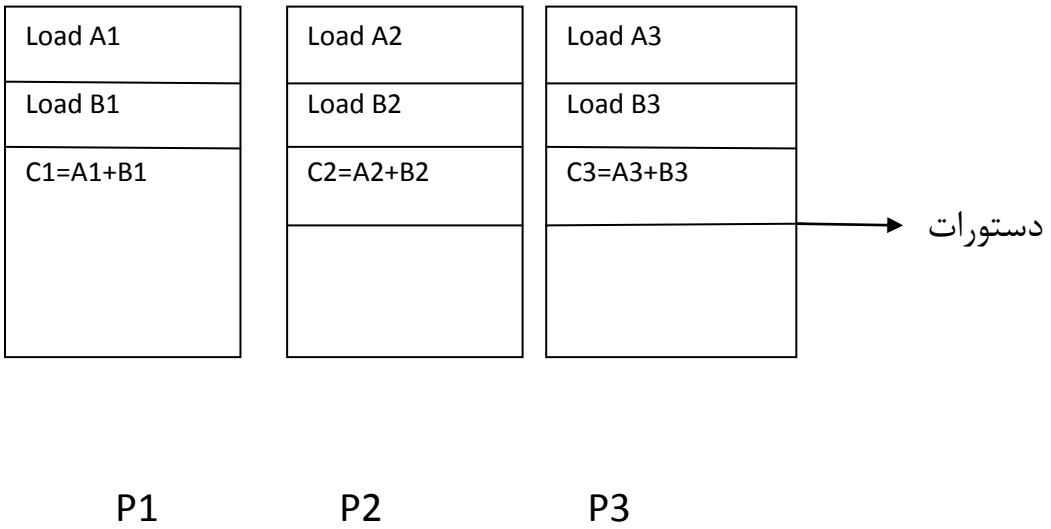
SIMD single instraction multiple data

MISD multiple instraction singele data

MIMD multiple instraction multiple data

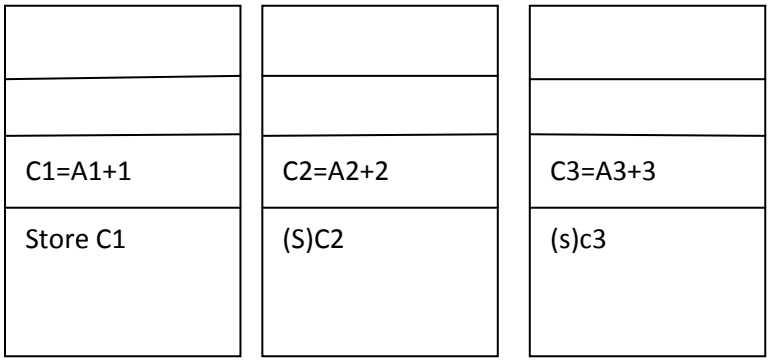
SISD : این معماری برای یک جریان دستوری روی یک جریان دیتا یا داده در طول پالس ساعت مورد استفاده قرار میگیرد. دستورات قطعی هستند وبه پردازنده ی دیگری وابسته نیست

SIMD: یکی از انواع کامپیوترهای موازی است همه واحد های پردازش یک دستور مشترک در طول هر پالس ساعت اجرا میکنند. اما هر CPU می تواند روی چند عنصر داده ای مختلف عمل کند.



MISD: یک جریان داده به چند واحد پردازش داده ارسال می شود هر واحد پردازش به طور مستقل با جریانهای دستور مستقل روی دیتا عمل میکند تا کنون تعداد محدودی کامپیوتری موازی با این روش ساخته شده.

از جمله کاربردهای این روش اعمال چند الگوریتم رمزگذاری باز کردن پیغام یک کد داده شده است.

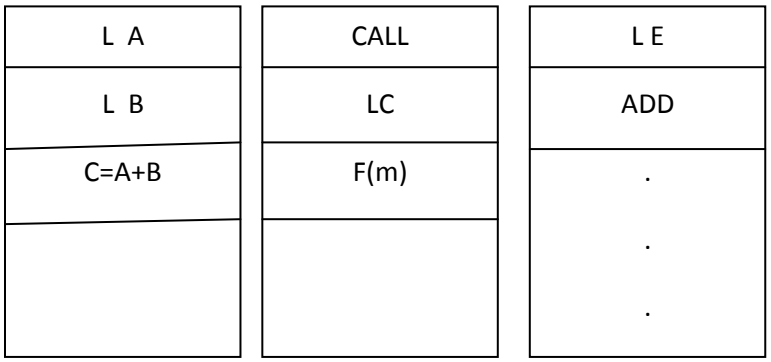


چند دستو روی دیتای واحد در چند پردازنده

نکته: همه ی این دستورات در یک پالس زمان (واحد)اجرا می شود.

MIMD: معمول ترین طرح کامپیوتری موازی است، هر پردازنده امکان اجرای چند جریان دستوری جداگانه روی چند جریان داده ی مختلف را دارد، عملیات اجرایی می تواند قطعی یا غیر قطعی باشد

(عملیات انجام می شود ولی خروجی ممکن است به پردازش دیگری وابسته باشد) ، کامپیوترهای موازی خوشه ای ، ابر کامپیوتر ها ، کامپیوترهای چند پردازنده ی SUP ، PCهای چند هسته ای امروزی از این معماری استفاده می کنند



نکته: سیستم های چند پردازنده به دو دسته تقسیم می شوند به طوری که اگر همه پردازنده ها به طور یکسان بتوانند تمام دستورات سیستم عامل را اجرا کنند ، به آنها سیستم چند پردازنده متقارن گفته می شود اما اگر بعضی از پردازنده ها دارای امتیاز بیشتر یا کمتر نسبت به سایرین باشند ، به آنها سیستم چند پردازنده ی نا متقارن گفته می شود.

کارایی نسبت به هزینه:

یک سیستم موازی با N پردازنده ی معمولی کارایی کمتری نسبت به یک پردازنده با سرعت N برابر دارد ، اما ساخت یک سیستم موازی داری قیمت پایین تری است ،

برای برنامه های که هم محاسبات فراوانی دارند و هم دارای محدودیت زمانی هستند و می توان انها را به N زیر برنامه (نخ) تقسیم کرد ، محاسبه موازی بهترین راه حل است.

الگوریتم ها:

نباید تصور شود که محاسبه ی موازی ، تنها محتاج به فراهم کردن سخت افزار مورد نیاز و اتصال درست آنهاست ، دستیابی به افزایش خطی سرعت ، بسیار مشگل است ، این مشگل ناشی از طبیعت ترتیبی بسیاری از الگوریتم هاست ، به طوری که قسمت های از یک الگوریتم غیر قابل موازی سازی است ، برای اثبات این حقیقت قانونی داریم به اسم آمدال Amdhl به این شکل ارائه می شود ، فرض کنید f=10% از یک الگوریتم که قابلیت موازی سازی ندارد ، اما بقیه ی الگوریتم به طور موازی توسط n=20 پردازنده اجرا می شود ، در این حالت سرعت اجرای این برنامه نسبت به زمانی که تنها روی یک پردازنده اجرا شود ۲۰ برابر نمی شود بلکه طبق رابطه Amdhl با ضریب افزایش میابد

سرعت ۷ برابر می شود $\frac{1}{f+\frac{1-f}{n}} = \frac{1}{1/10+\frac{1-1/10}{20}} \cong 7$

نه ۲۰برابر که همان تعداد پردازنده هاس

جلسه دوم پردازش موازی ۹۵/۸/۱۹

قانون مور: عملکرد ریز پردازنده ها با فاکتور ۲(یعنی دوبرابر) روبه رشد است .

در جهت اصلاح قانون مور این زمان به ۱۸ ماه تغییر کرده که معادله ۶۰٪ در سال است .

این رشد ناشی از ترکیب این دو عامل است .

۱-افزایش پیچیدگی تراشه های vlsi و پیش بینی افزایش به حدود۱۰میلیون ترانزیستور در یک چیپ: تراشه

۲-بهبود در معماری ها مانند حافظه ها (بافر و cash)،چند نخی، خطوط لوله عمیق ،

نکته: قانون مور در ۱۹۶۵ با مضمون دو برابر شدن پیشرفت وپیچیدگی چیپ (تراشه) در هر ۱۸ ماه براساس تعداد داده ی کم ، تدوین شده است. در این قانون دو مسله ی IPS و FLOPS

IPS: تعداد دستورالعمل های اجرایی در هر ثانیه توسط CPU

FLOPS: عملیات ممیزی شناور در هر ثانیه

محدودیت های قانون مور:

۱-محدودیت سرعت نور: سرعت نور برابر 3×10^8 است که در فیبر نوری استفاده می شود اما در تراشه ها از سرعت نور استفاده نمی شود بلکه از انتشار سیگنال استفاده میکنیم. که در نهایت $1/3$ سرعت نور است که همان 10^8 است.

مساحت

بر اساس فرمول $T=\frac{V}{d}$ اگر قطر تراشه ۱ سانتی متر باشد.

سرعت انتشار

$$T=\frac{V}{d}=\frac{1\text{cm}}{1\times10^8}=\frac{10^{-2}}{10^8}=10^{-10}=0.1\times10^9=0.1\,ns$$

یعنی در یک دهم (0.1) نانو ثانیه یک دستورالعمل اجرا می شود.برای رفع این محدودیت باید مساحت یا قطر تراشه ها را کم کرد. بنابراین باید به فکر تکنیک های موازی سازی بود (که این کم کردن زیر ۱۸ ماه امکانش نیست)

زیرشاخه های MIMD از نظریه FLYNN:

MIMD شامل چهار زیر کلاس

1-حافظه عمومی متغیر ها اشتراکی(GMSVD)

Global memory sheired variable

2- حافظه عمومی روش انتشار پیغام:(Gmmp)

Global memory message passing

3-توزیع شده – حافظه اشتراکی (Dmsv)

Distri buted memory shared variable

4- حافظه بصورت توزیع شده بین پروسه ها تقسیم میشود ارتباط پروسه ها از طریق انتشار پیام است(Dmmp)

Distri buted memory message passing

موانع پردازش موازی:

۱ -قانون grosh:هزینه تحقیقاتی p تا پردازنده معادل هزینه ی p^2 در یک سیستم یک پردازنده است.

۲ -قانون minsky: افزایش سرعت متناسب با $\log_2 p$ می باشد pتعداد پردازنده هاست.

۳ -فناوری های IC: در هر ۵سال سرعت ۱۰ برابر می شود.

۴ -قانون Amdhl(آمدال): کد یا دستورات غیر موازی سرعت را به شدت محدود می کند.

Sهمان کارایی cpuاست $S=\frac{1}{f+\frac{1-f}{p}}$

نهایتا به این نتیجه میرسیم که فقط موازی سازی فقط به تعداد پردازنده ها بستگی ندارد بلکه به برنامه یا کدی که قرار است اجرا شود نیز بستگی دارد.

ارزیابی سیستم های پردازش موازی : به منظور ارزیابی سیستم های با پردازش موازی یک سری پارامترها تعریف شده است.

۱ -اولین پارمترp است که معادل تعداد پردازنده هاست

۲ -W(p):تعداد عملیاتی که پردازنده pام انجام می دهد

۳ -T(p):زمان اجرای عملیات توسط pتا پردازنده ، اگر$p = 1 \leftarrow$ w(1)=T(1)

۴ -S(P): سرعت است که از قانون آمدال پیروی می کندS(p)=$\frac{T(1)}{T(P)}$

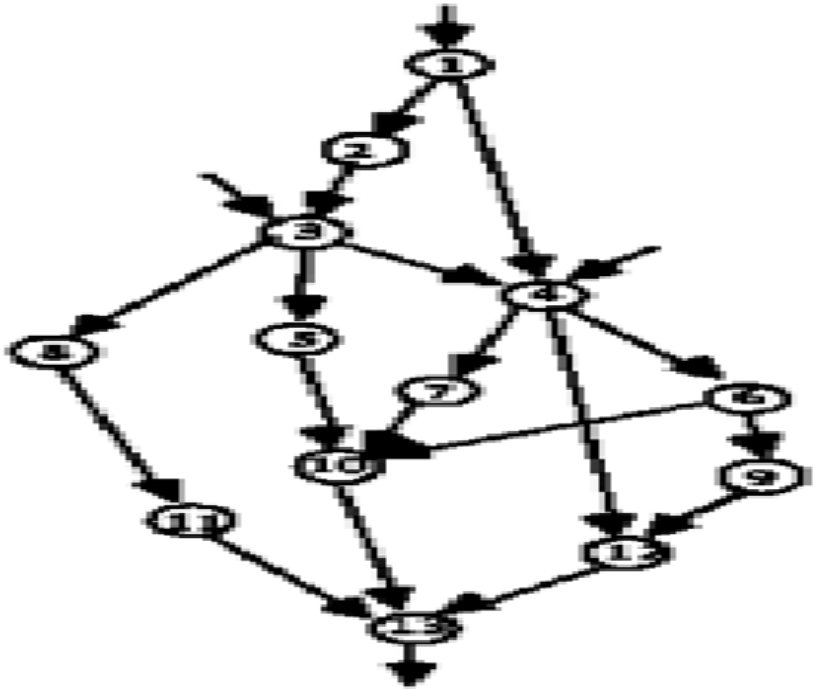
۵ -E(P):بهروری E(p)= $\frac{T(1)}{P \times T(P)}$

۶ -R(p): سخت افزاری که به صورت اضافی اسفاده شده:R(p)=$\frac{W(p)}{W(1)}$

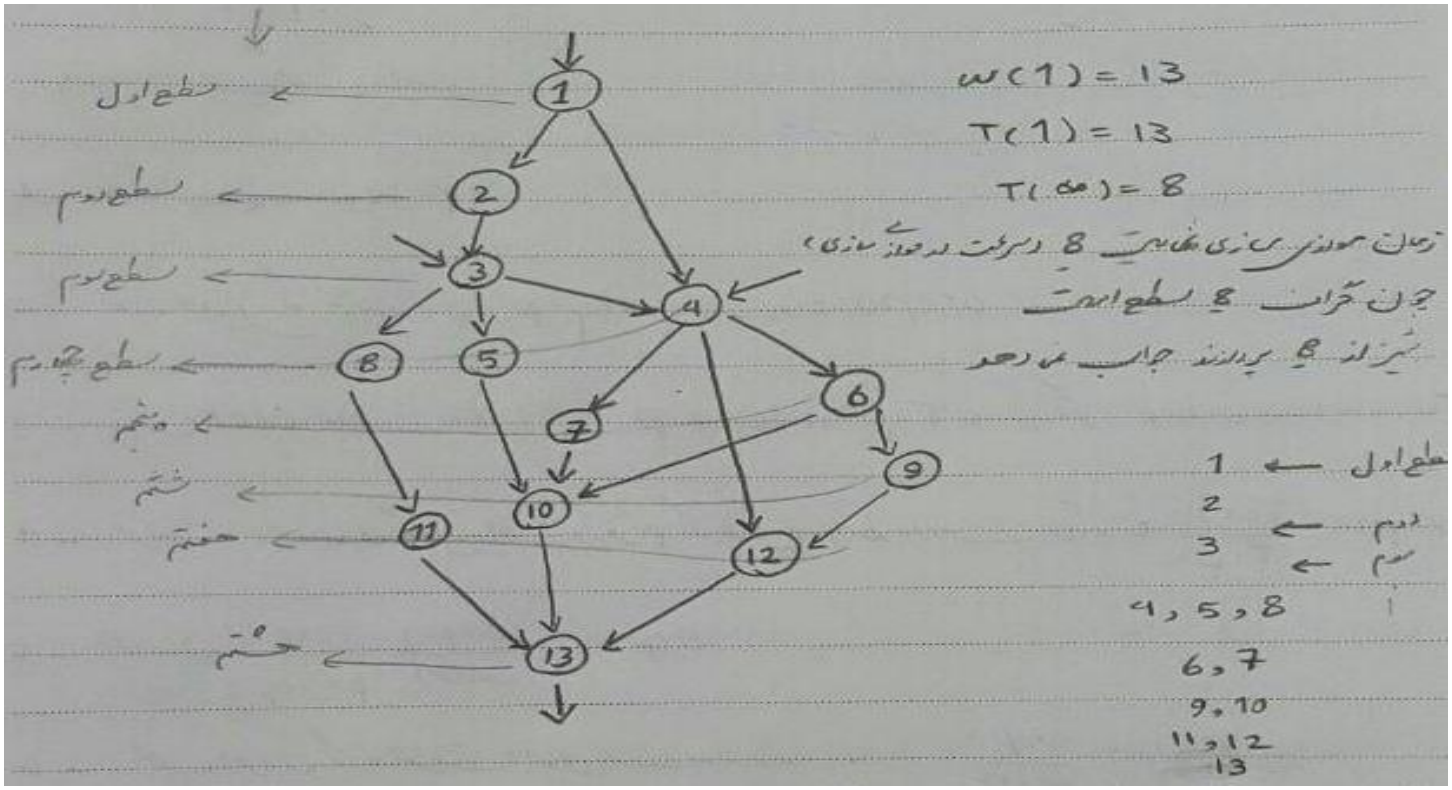
۷ -Q(p) :کیفیت: Q(p)= $\frac{T^3\ (1)}{P \times T^2(\mathrm{p}) \times \mathrm{W}(\mathrm{p})}$

مثال:در موارد ارزیابی سیستم ها

گراف زیر داده شده است اگر زمان اجرای هر پروسه ۱ واحد باشد پارمترهای W(1) و T(1) و t(∞)را

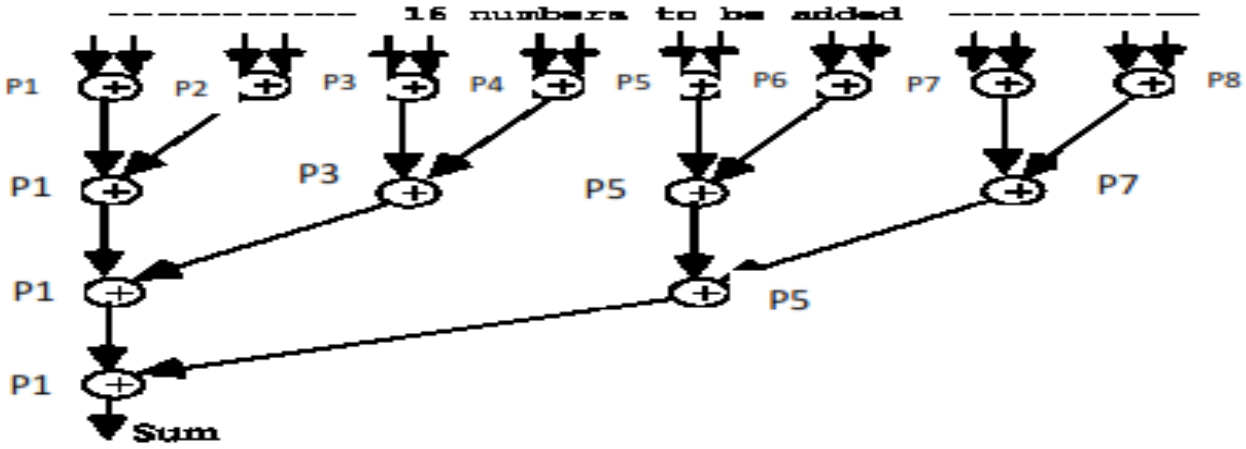


محاسبه کنید



مثال دوم: جمع ۱۶ عدد دریک سیستم ۸پردازنده ای

الف: اگر هزینه ارتباطی بین پردازنده ها را 0(صفر)در نظر بگیریم ، پارامترهای Q(p),R(p),E(P),S(p),W(P),T(p),W(1),T(1)را محاسبه کنید.



T(1)=15 تعدادعملیات ها

T(1)=w(1)=15

T(8)=4 تعداد سطوح گراف

W(8)=15 تعدادعملیات)تعداد کارهای که توسط پردازنده

$S(p)=\frac{t(1)}{T(P)}=\frac{15}{4}\cong\frac{3}{75}$

s(8)=

$e(P)=\frac{w(P)}{p*tip}=\frac{15}{8\times4}=47\%$

47%درصد یعنی ۵۳٪ از پردازنده ها بیکار هستند

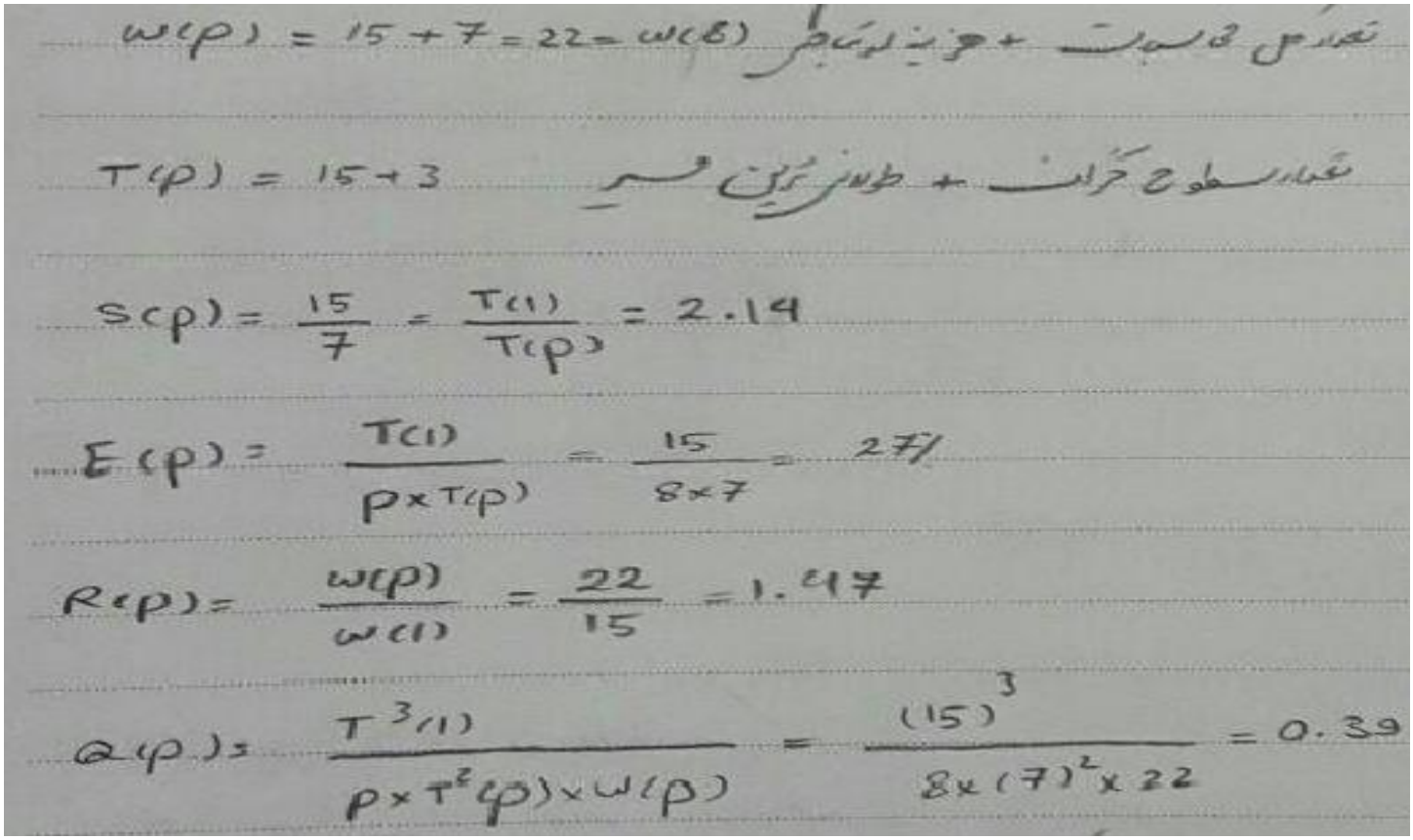
$R(p)=\frac{W(p)}{w(1)}=\frac{15}{15}=1$ اگرR(p)=1باشد نیاز به سخت افزار اضافی نداریم

$Q(p)=\frac{t(1)}{P\times T^2(P)\times W(p)}=\frac{15^3}{8\times4^2\times15}=1/76$

نکته:اگر هزینه ارتباطی بین پردازنده ها در صورت سوال لحاظ شود باید هزینه ی طولانی ترین مسیر تا نتیجه نهایی را با تعداد سطوح گراف اضافه کنید

نکته:برای ۸ پردازنده (به عنوان مثال) اگر هزینه ارتباطی بین پردازنده ها را ۱ در نظر بگیریم این هزینه در کل ۷ می شود.

بنابراین مثال قبل به هزینه ارتباطی ۱ بین پردازنده ها به این شکل تغییر می کند



هزینه ارتباط همیشه در حالت سری محاسبه میکنیم

نگاهی سریع به برخی از الگوریتم های موازی:

5نوع محاسبه موازی داریم:

۱ -محاسبات زنجیره ای Semi group زمان اجرا=تعدادسطوح=$\log_2 n$

۲ -**محاسبات موازی با پیشوندی prefix** زمان اجرا=تعدادسطوح=$\log_2 n$

۳ -**محاسبات بسته ای pocket rating**

۴ –محاسبات انتشاری broed costing

۵ -مرتب سازی sorting

نکته: محاسبات موازی همانند زنجیرایی است با این تفاوت که نتایج میانی هم نیاز است.

وحدقل به $\log_2 n$ برای محاسبه نیاز داریم.

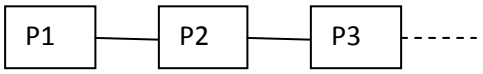
معماری پردازنده ها :

در پردازنده هایی که به صورت آرایه هایی خطی به هم وصل هستند دو پارامتر تعریف می شود

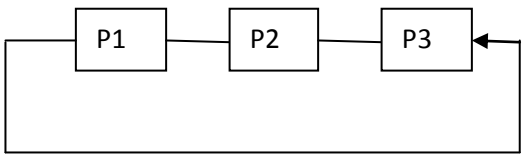
D: **قطر پردازنده:** تفاوت بین طولانی ترین و کوتاه ترین مسیر بین یک زوج پردازنده

در مدل آرایه ای خطی معمولی

حداکثر درجه هر گره (D): حد اکثر پیوندها یا کانال های ارتباطی برای یک پردازنده



در آرایه خطی معمولی d=D=2



در آرایه حلقوی $D=\left[\frac{P}{2}\right]=\frac{8}{2}=4$

جلسه سوم پردازش موازی:

پردازنده هایی که به صورت درخت دودویی به هم متصل ه‌ستند:

یک درخت باینری نه لزوماً کامل و نه لزوماً کاملاً متوازن ، می باشد.

d(حداکثر درجه گره ها):همانند liner، تعداد انشعابات خروجی از هر نود

$D: z\lceil \log_p 2 \rceil$

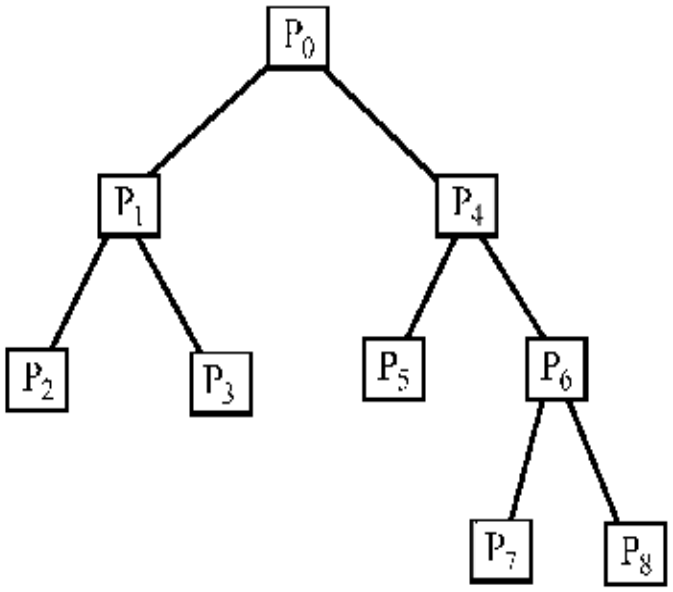


Fig. 2.3 A balanced (but incomplete) binary tree of nine processors.

d:3

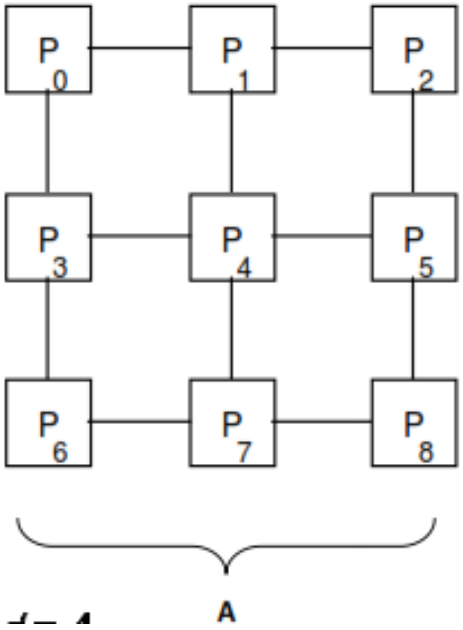
D:2 $[\log_2 9]$

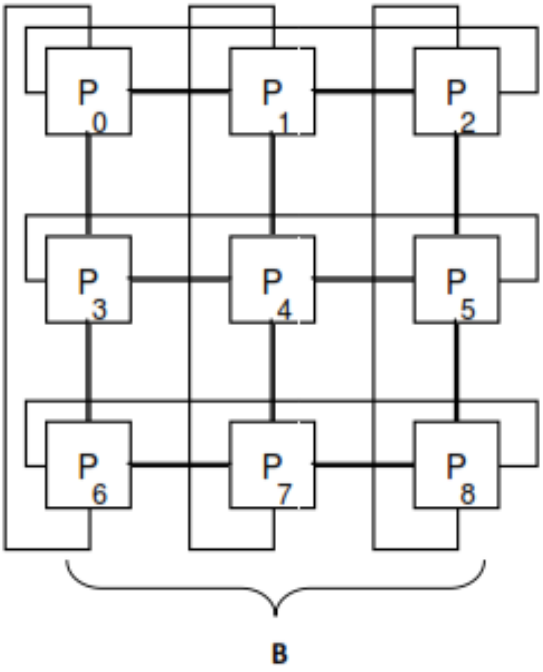
پردازنده هایی که به صورت Mesh دو بعدی به هم متصل هستند:

پردازنده ها به شکل Mesh دو بعدی می توانند یا به شکل سطر و ستون مانند شکل (A) ویا به شکل ستونی از ابتدایی ترین پردازنده به انتهای آن ستون و با شکل سطری (از ابتدایترین پردازنده به انتهای آن متصل شوند.

d: حداکثر درجه نودها

$$D_A: 2\sqrt{p-2} \qquad D_B: \sqrt{p}$$





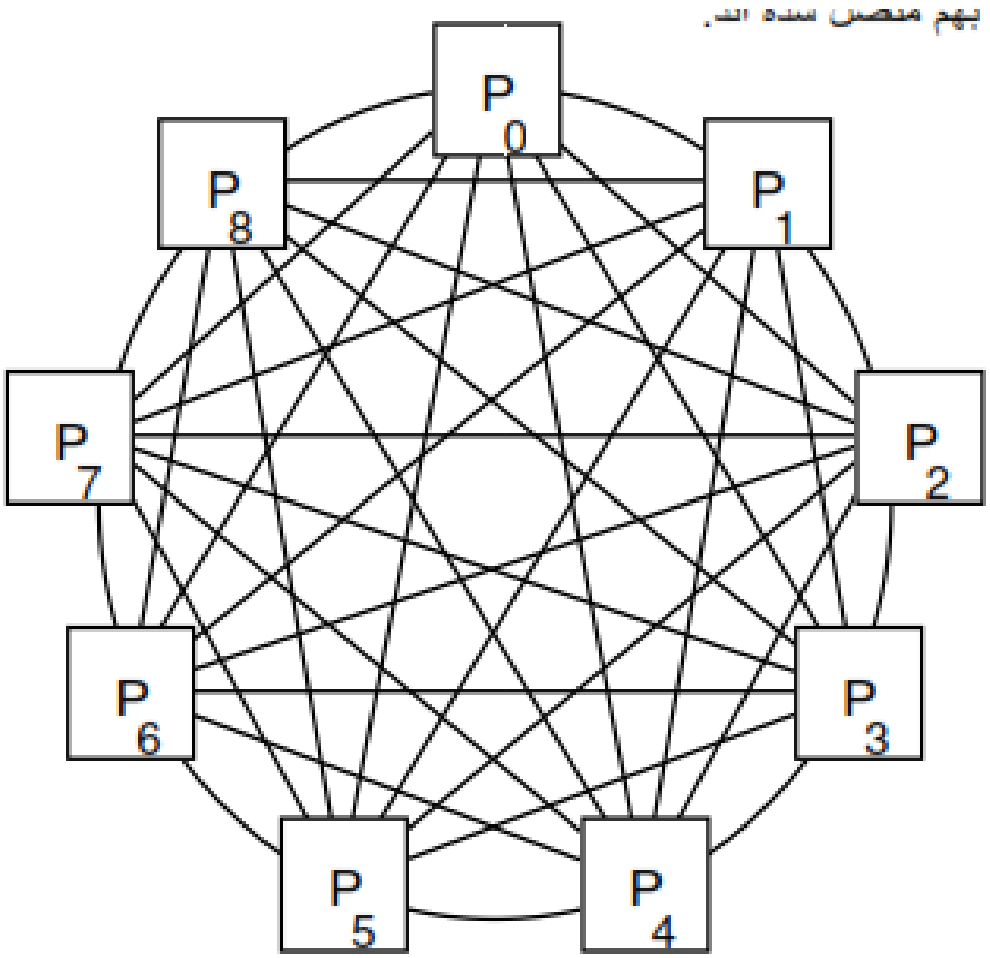
نکته:

ضریب معماری روع B به نوع A ، این است که مقدار پارمتر Dکاهش یافته است.

پردازنده های که به صورت معماری حافظه اشتراکی به هم متصلند:

d:8 حداکثر درجه گره ها

D:1 چون به هم گره ها مسیر دادیم



ترکیب ۴ معماری موازی با ۵ الگوریتم موازی:

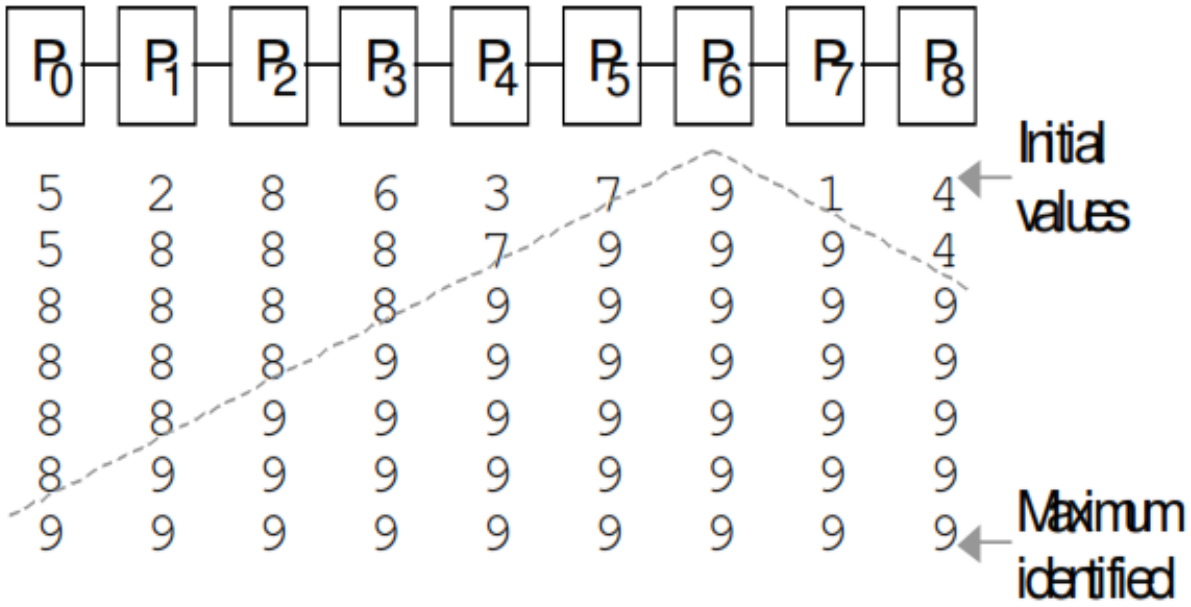
۱ -الگوریتم هایی برای معماری خطی:

صفت:الگوریتم semigroup (زنجیره ای) فقط نتیجه ی پایانی مهم است، نتایج میانی مهم نیست

مثال: فرض کنید یک معماری آرایه ی خطی با ۹پردازنده داریم . طبق شکل در هر پردازنده ۱ عدد ذخیره شده است ، هدف محاسبه حاصل جمع (max) این عداد است.

طبق الگوریتم زنجیره ایی روی معماری خطی انجام می شود.

هر پردازند در هر مرحله مقدار ماکزیمم خود را به دو همسایه مجاور خود عمال می کند هر پردازنده با دریافت مقادیر پردازندهای مجاور ماکزیمم رابه صورت رابطه ای بدست می آوریم.



الگوریتم زمانی خاتمه می یابد که هر پردازنده از همسایه های خود دو عدد یکسان دریافت کند یا همه پردازنده ها عدد یکسان دریافت کنند.

ب: الگوریتم prefix با موازی برای یک معماری خطی:

در این الگوریتم جواب های میانی مورد نیاز است .

مثال: فرض کنید یک معماری آرایه ای خطی با ۹ پردازنده داریم و در هر پردازنده یک عدد ذخیر شده است ، هدف محاسبه ی حاصل جمع این اعداد است.

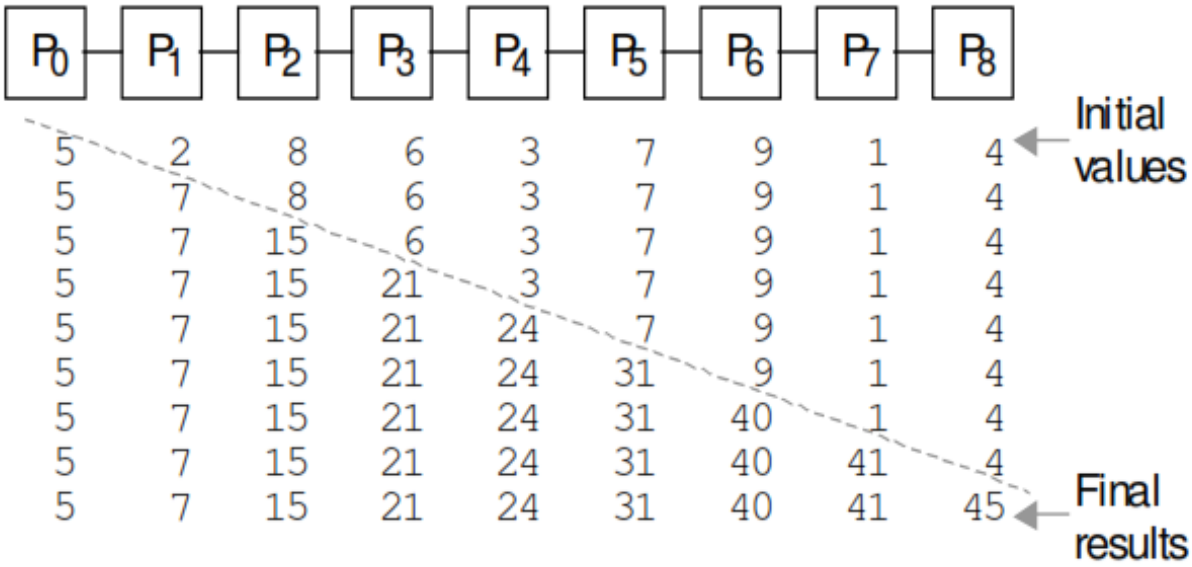
قاعده اول : در هر مرحله یک پردازنده مقدار خود را با مقدار پردازنده سمت چپ خود جمع کرد و در خود ذخیره میکند.

قاعده دوم: در هر مرحل فقط یک پردازنده عمل جمع را انجام می دهد وباقی پردازنده ها منتظر می میانند

قاعده سوم: عمل جمع از اولین پردازنده آغاز و در مرحله بعدی تاثیر خو را روی پردازنده بعدی به عنوان ورودی جمع میگذارد الگوریتم در آخرین پردازنده به پایان میرسد.

نکته: در این مثال تفاوت به لحاظ زمان اجرا بین یک سیستم دو پردازنده ای و یک پردازنده ای وجود ندارد.

مثال:



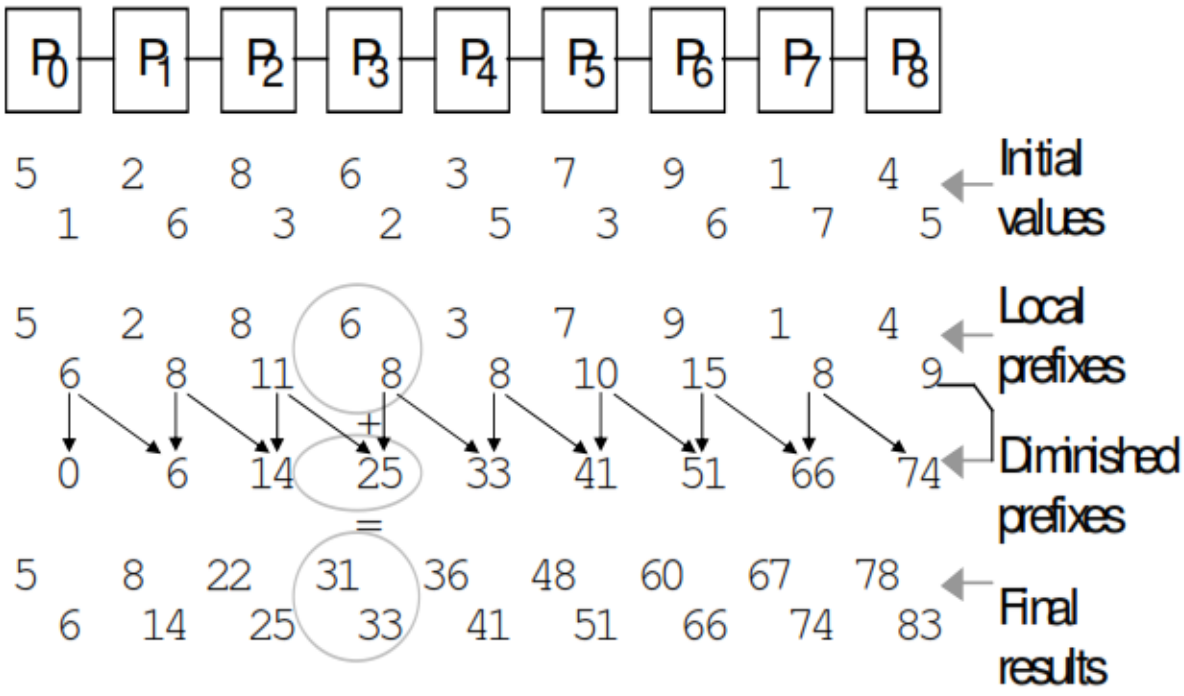
در نهایت عدد ۴۵ که جمع هم است در پردازنده ی آخر به ما نشان می دهد.

موازی سازی در این مثال نیست .

مثال: فرض کنید یک معماری آرایه ای خطی با ۹ پردازنده داریم و در هر پردازنده دو عدد ذخیر شده است ، هدف محاسبه ی حاصل جمع این اعداد است.

- محاسبه مجموعه داد ها در هر پردازنده انجام می شود
- عملیات جمع به صورت prefix انجام می گردد
- در نهایت حاصل جمع های محاسبه شده در مرحل قبل با هم جمع می شود .
- مطابق شکل در مرحله ی ....local هر پردازنده به صورت همزمان در مقدار خود را جمع می کند.
- در مرحله ای بعدی diminish prefixحاصل جمع های محاسبه شده در مرحله قبل به صورت سری و غیر همزمان با هم جمع می شود، در نهایت final result ، حاصل جمع های محاسبه شده در مرحله قبلی با هم جمع می شوند. (چون جواب قبلی هم مهم می باشد)

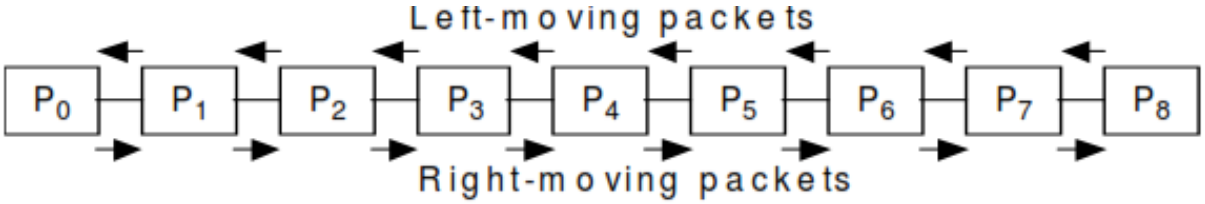
هر پردازنده باید صبر کند تا نتیجه را از پردازنده قبلی گیرد . موازی سازی نداریم.



ج: الگوریتم مسیر یابی بسته ای packet routing برای (روی) آرایه خطی liner:

پردازنده ی pi می خواهد بسته ای را به pj ارسال کند . محاسبه (j-i) برای تعیین فاصله و جهت حرکت که به آن routing tag می گوییم در نظر گرفته می شود.

اگر این مقدار مثبت باشد حرکت به سمت راست و اگر منفی باشد حرکت به سمت چپ است



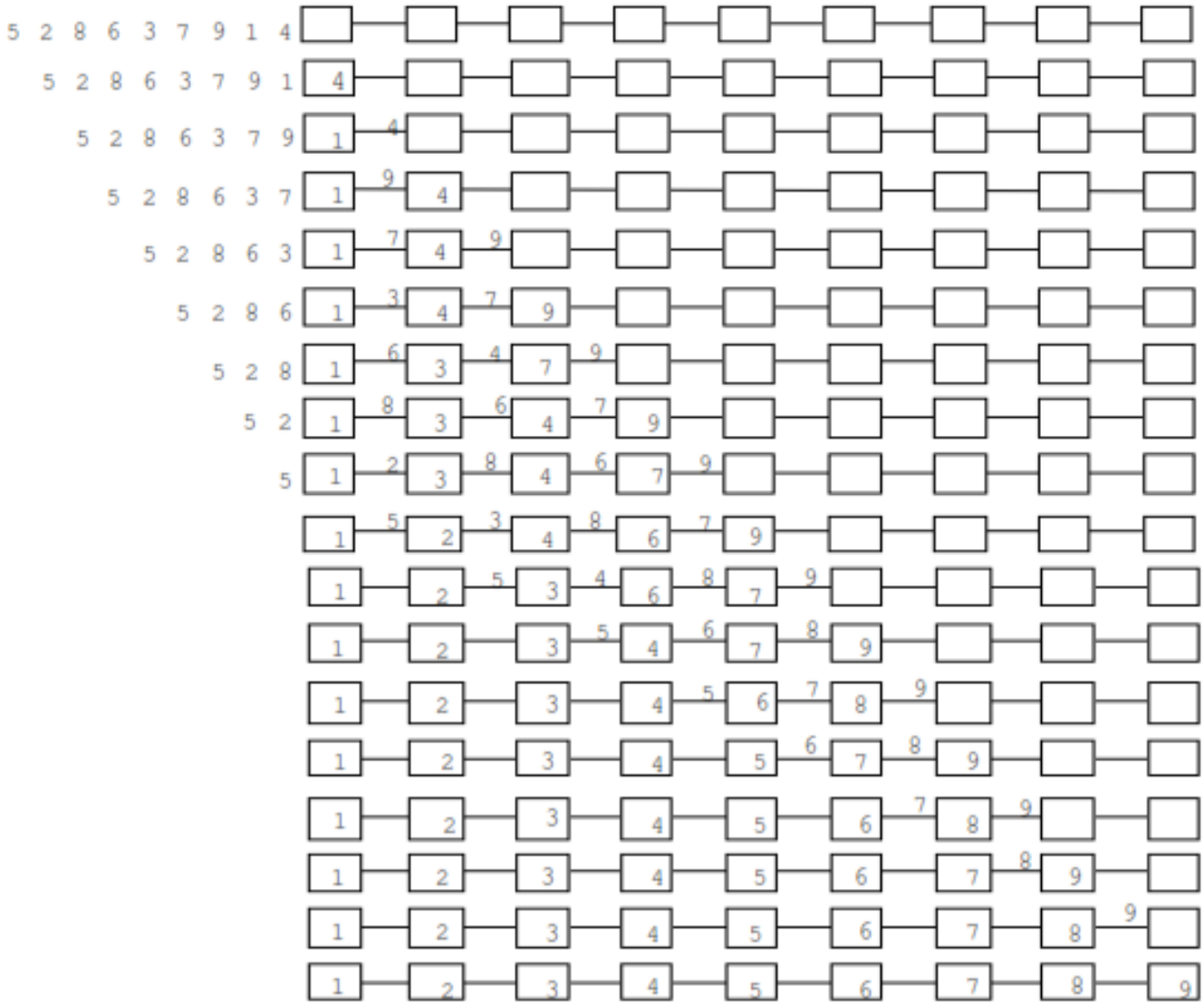
د: مسیر یابی انتشاری (broad costing) روی یک معماری خطی :

در این حالت اگر پردازنده ی pi بخواهد بسته ای را به تمام پردازند ها ارسال کند. آن پیام را به صورت R B cast (a) به همسایه سمت راست و پیام L b cost (a) به همسایه ی چپ خود ارسال می کند .

و: الگوریتم مرتب سازی sorting روی معماری liner (خطی):

مقدار اولیه پردازند ها تهی (∞) یا صفر است.

هر پردازنده یک مقدار کلید از سمت چپ دریافت می کند و آنرا با مقدار ذخیره شده در ریجستر (register) داخلی خود مقایسه می کند . مقدار کوچکتر (کوچکترین) را نگه داشت و بزگتر را به پردازنده ی سمت راست خود انتقال می دهد.(الگوریتم درجی)



مثال: مرتب سازی add / even (زوج و فرد):

دارای دو مرحله زوج و فرد

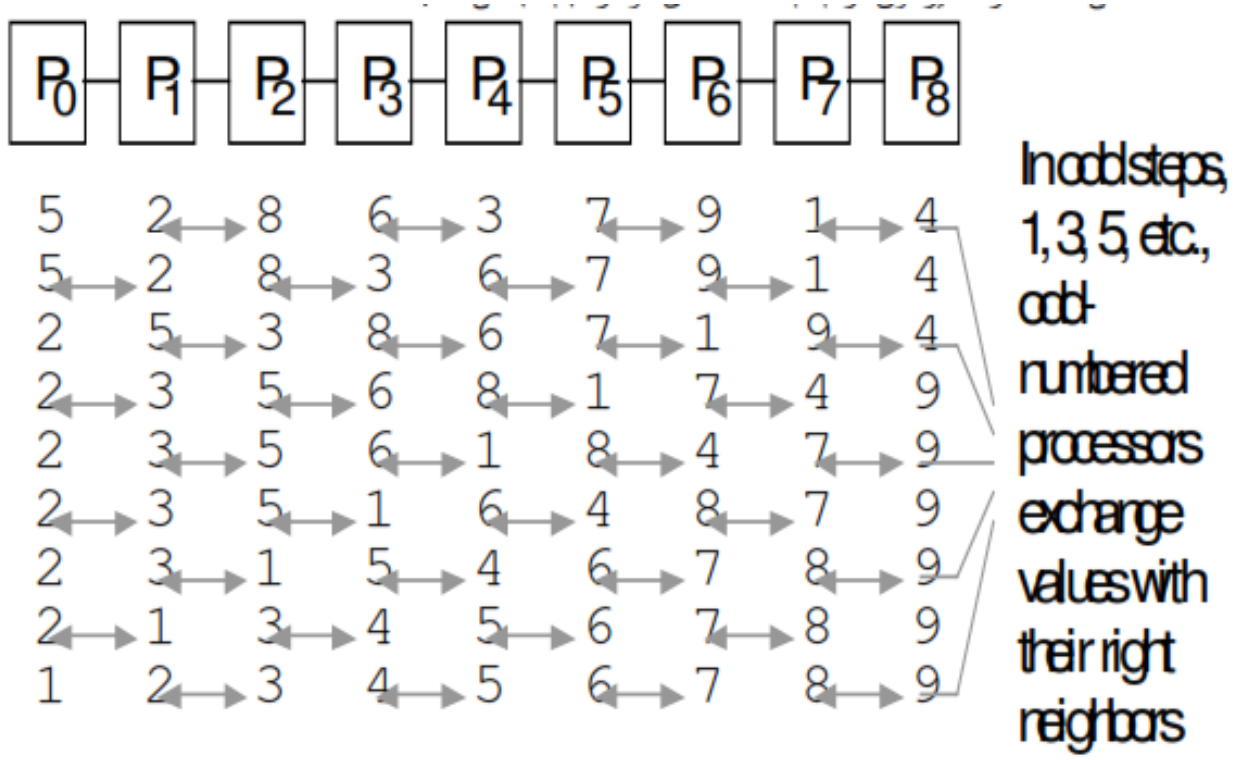
مرحله add (فرد): در این مرحله پردازنده های add (فرد) مقادیر خود را با همسایه های زوج سمت راست مقایسه میکنند. اگر مقادیر فوق مرتب نباشند : داده های خود را جابه جا میکنند

مرحله even namber (عداد زوج): در این مرحله پردازنده های زوج مقادیر خود را با همسایه های فرد سمت راست مقایسه می کنند ، اگر مقادیر مرتب نباشند . داده های خود را جابجا می کنند.

مثلاً در مرحله ی add namber ، ابتدا به طور همزمان p1 مقدار خود را با p2 و p3 مقدار خود را با p4 ،p5 با مقدارp6 ، p7مقدار خود را با p8 مقایسه میکند و اگر لازم باشد مقادیر جابه جا می شود . سپس در مرحله even

namber : p مقدار خود را با p1 و p2 مقدار خود را با p3 و p4 مقذار خود را با p5و p6 مقدار خود رابا p7 مقایسه می کند. و اگر لازم باشد مقادیر جابه جا می شود.

الگوریتم تا زمانی ادامه می یابد که هیچ جابه جایی صورت نگیرد .



Gpu: از تعداد نسبتاًزیادی پردازنده ساده و کوچک که به آنها اصطلاحاً thread یا نخ گفته می شود ، ساخته می شود . هر کدام از این نخ ها می تواند یک ریز پیکسل را درون پالس ساعت پردازش کند . قبل از معرفی (cuda) تعداد کمی از محققین استفاده از آنرا آغاز کردند. اما به دلیل مشگلات زیاد کار با آن متوقف شد.

مهترین مشگل این بود که تنها راه بر قراری ارتباط با کارت گرافیک از طریق نرم افزارهای خاصی مثل open gl ، directy بود ، و این واسطه ها فقط و فقط مختص پیکسل بودن ، با معرفی (cuda) اکثر مشکلات بر طرف شد، cuda به این معناست که دستگاه ای سخت افزاری جانبی مثل cpu داشته باشند وبتوانند بدون نیاز به cpu پردازش را انجام بدهند. در واقع با اینکار بار محاسباتی از روی دوش cpu برداشته شد ، قدرت پردازش gpu از cpu بالاتر می باشد ، فلسفه اینکار این است که یک کار گرافیکی سنگین در (vga) یا همان کارت گرافیک با استفاده از cpu های تعبیه شد روی آن به صورت موازی و با سرعت بیشتر از cpu اصلی عملیات انجام می دهند . البته برنامه نویس خودش تعیین می کند که پردازش در هر مرحله تئسط cpu انجام شود یا cuda با همان کارت گرافیک یک cpu در مرحله اول با gpu در ارتباط است . داده های تحت پردازش از حافظه ی اصلی به حافظه gpu کپی می شود . دستور پردازش توسط cpu به gpu داده می شود ، برنامه ها به صورت موازی در هر نخ اجرا می شوند ، (درونgpu) نتایج از پردازند ها (نخ) به حافظ gpu منتقل می شود . و در نهایت از gpu به حافظ ی اصلی منتقل می شود

جلسه چهارم

الگوریتم های موازی روی درخت دودویی:

در الگوریتم هایی که برای پردازنده هایی به شکل درخت باینری لحاظ می شود فرض بر آن است که داده ها در برگ درخت ذخیره می شوند وپردازنده های غیر برگ تنها در محاسبات شرکت می کنند اما عناصر داده را در خود نگه نمی دارند.

محاسبات زنجیره ایی semi group بر روی درخت باینری :

در ابتدا داده ها در گروه های برگ ذخیره شدند نتایج محاسبات در node یا گره های غیر برگ انجام می شود . ونتیجه نهایی در node ریشه یا (root) ذخیره می شود.

نکته:

این محاسبه و اعلام نتیجه در گره root یا ریشه به $\log_2 p$ زمان نیاز دارد

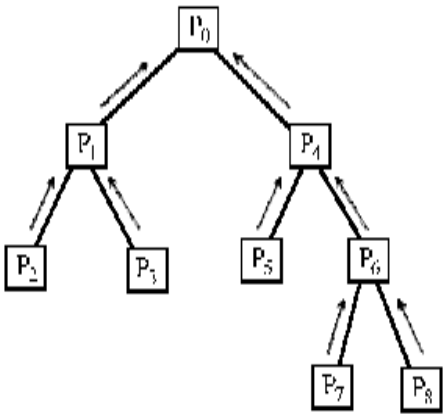
P تعداد پردازنده هاست

نکته:

حال اگر گره root بخواهد نتیجه را به سایر پردازنده ها اعلام کند به $\log_2 p$ زمان نیاز دارد .

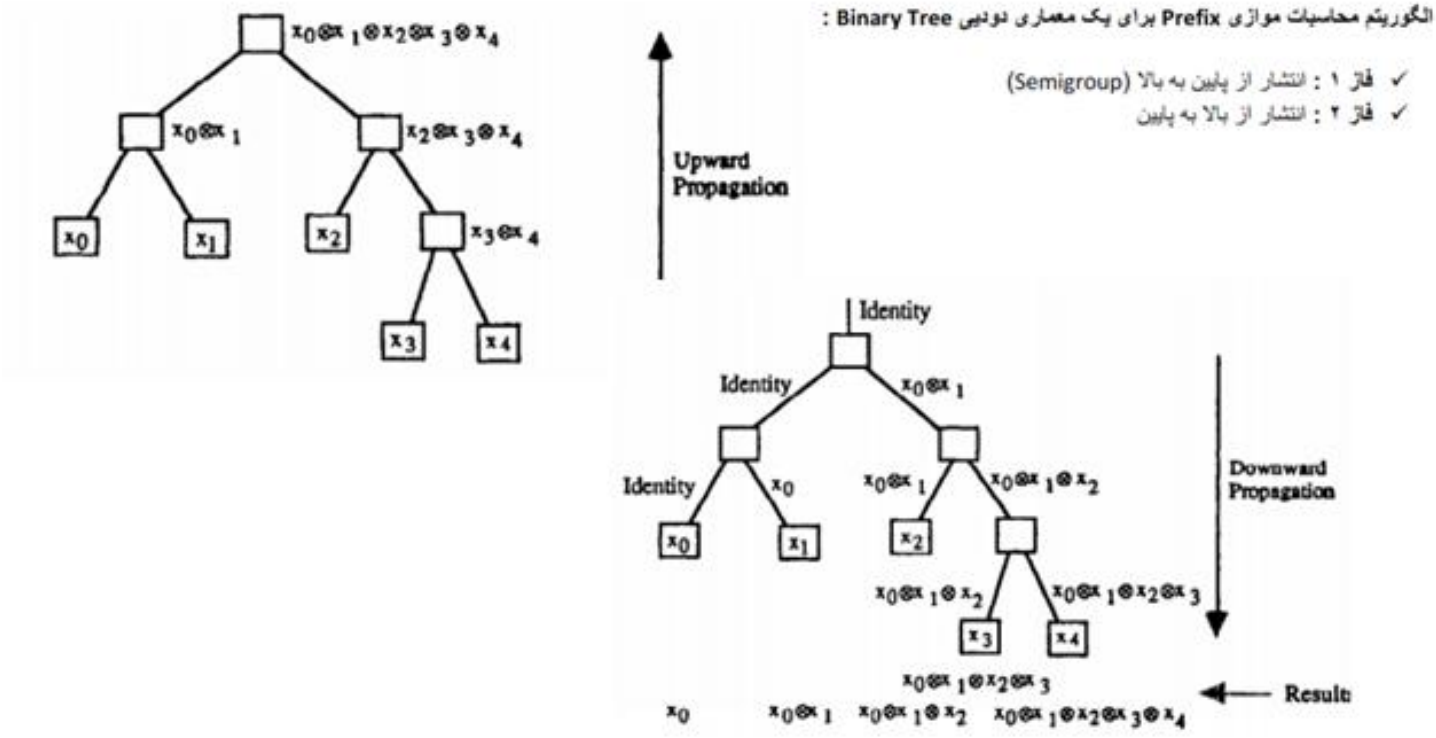
پس در مجموع زمان محاسبه و انتشار (مرحله ی ۲) به $\log_2 p$ دو زمان نیاز دارد.

الگوریتم Semigroup برای یک معماری دودیی Binary Tree :



الگوریتم محاسبات موازی برای درخت باینری و پرفکس نتایج میانی مهم است .

در دو فاز این الگوریتم انجام می شود. در انتشار از پایین به بالا همانند محاسبات زنجیره ایی عمل می کند در محاسبه ی بالا به پایین به این شکل است که هر پدازنده مقداری که از فرزنده چپ خود دریافت می کند به خاطر می سپارد و همچنین مقداری را که از کره پدرش دریافت می کند به سمت چپ انتقال می دهد و فرزنده چپ با فرزنده سمت راست ترکیب محاسباتی می شوند . این عمل تا node آخر سمت راست ترین برگ ادامه می یابد .



زمان محاسبه این الگوریتم برابر با $2\lceil \log_2 p \rceil$ p همان تعداد پردازنده هاست

نکته:

Nodeهای غیر برگ نتایج را نگه می دارند در روش موازی اما در روش قبلی node های غیر برگ مقادیر را نگه نمی دارند.

الگوریتم مسیر یابی packet routing بر روی درخت دودویی :

هدف:

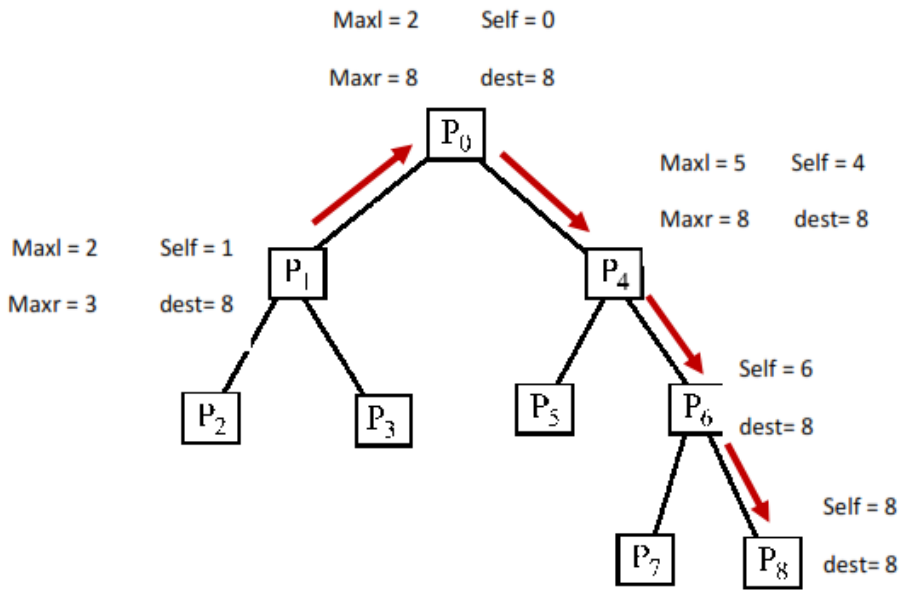
ارسال بسته از پردازندهی iام به پردازنده ی jام می باشد

شرح الگوریتم:

اگر گره ای مقصد همان گره ی مبدا است الگوریتم پایان می یابد در غیر این صورت اگر شماره ی گره ی مقصد کوچیکتر از گره ی مبدا بود و یا شماره ی گره ی مقصد بزرگ تر از بزرگ ترین شماره در سمت راست باشد حرکت به سمت بالا ادامه می یابد در غیر این صورت اگر شماره گره ی مقصد کوچیکتر یا مساوی بزگترین شماره گره در سمت چپ باشد حرکت به سمت چپ منتقل می شود در غیر این صورت حرکت به سمت راست است .

الگوریتم مسیریابی بسته Packet Routing برای یک معماری دودیی Binary Tree :

```
if dest = self
then remove the packet {done}
else if dest < self or dest > maxr
   then route upward
   else if dest ≤ maxl
      then route leftward
      else route rightward
      endif
   endif
endif
```



۴-الگوریتم broadcasting (همه پخشی) بر روی درخت دودویی:

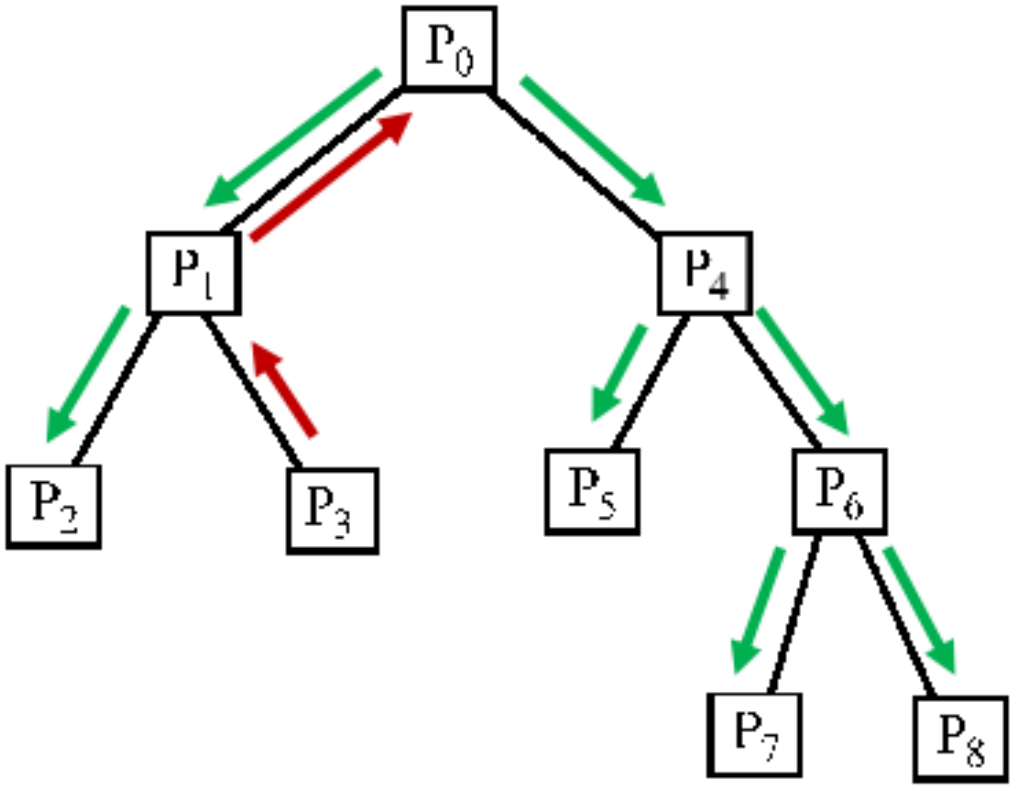
پردازنده ای iام می خواهد یک بسته را به تمام پردازنده ها ارسال کند. ابتدا آن را به سمت ریشه فرستاده و سپس ریشه ی آن را به همه پردازنده ها پخش می کند . یک بار بالا به پایین ویک بار پایین به بالا

نکته:

زمان حرکت دیتا از پایین بهسمت ریشه و سپس انتشار از ریشه به سمت کل پردازنده ها (گره ها) از $2\log_2 p$

کمتر می شود و احتمالا از درجه $\log \log_2 p$ خواهد بود

الگوریتم انتشاری بسته Brodcasting برای یک معماری دودیی Binary Tree :



۵-الگوریتم مرتب سازی sort بر روی درخت دودویی:

الگوریتم مرتب سازی Sort برای یک معماری دودیی Binary Tree :

```
if you have 2 items
then do nothing
else if you have 1 item that came from the left (right)
   then get the smaller item from the right (left) child
   else get the smaller item from each child
   endif
endif
```

در ابتدا در نود برگ یک داده دارد و گره های دیگر خالی هستند

هر گره ی داخلی ظرفیت ذخیره سازی ۲ داده را دارد عملیات پایین به بالا از چپ به راست انجام میشود

نکته:

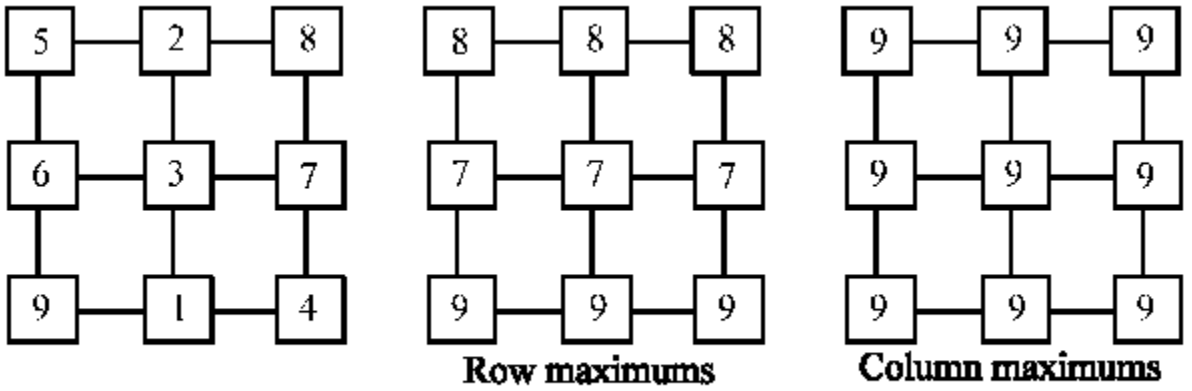
معمولا خواندن درخت به صورت سطری است . زیر درخت چپ زیر درخت راست الویت با چپ است

جلسه پنجم

۱ -الگوریتم زنجیره ایی بر روی مش دو بعدی: semi group

در مثال پیدا کردن ماکزیمم اعداد پردازنده ها در هر سطر به صورت موازی ماکزیمم رقم خود را اعلام می کنند . سپس پردازنده ها در هر ستون به صورت موازی نتایج خود را اعلام می کنند در نهایت اعداد ذخیره شده در تمام پردازنده ها نشان دهنده ماکزیمم اعداد است.

الگوریتم Semigroup برای یک معماری مش دو بعدی 2D Mesh :



Finding the max value on a 2D mesh.

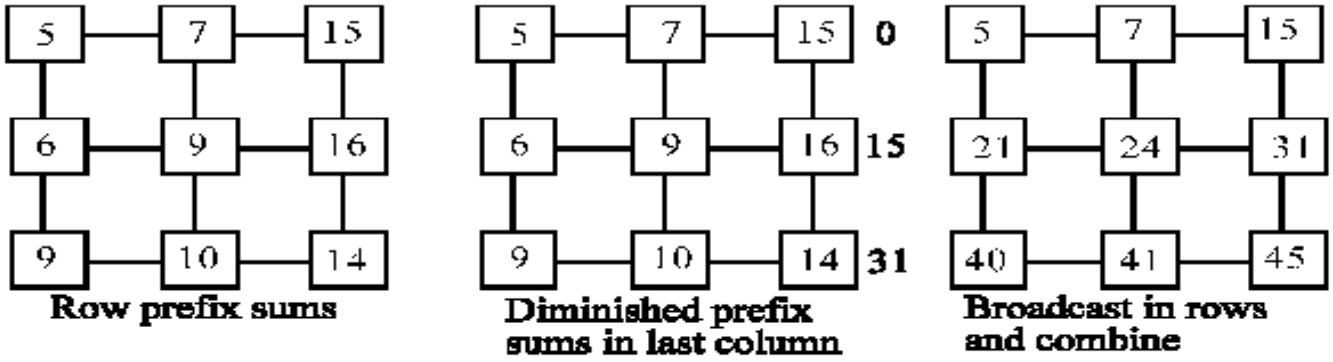
در فاز دوم ستون آخر : ۱۵ به پردازنده ی بعدی می رود که می شود ۳۱و۳۱ به پردازنده بعدی می رود می شود ۴۵

در فاز سوم حالت broad casting دارد نتیجه ی ستون نهایی در سطر پخش می شود .

حاصل جمع اعداد ذخیره شده در ۹ پردازنده:

در فاز اول پردازنده های هر سطر مجموعه نهایی خود را در پردازنده ی آخر هر سطر ذخیره می کنند. سپس ستون سمت راست (سمت راست ترین ستون) به شکل prefix کاهش یافته عمل می کند به این شکل که نتیجه ی هر پردازنده در آن ستون برابر با نتیجه ذخیره شده در پردازنده قبلی به اضافه ای داده ذخیره شده در خودش می باشد ،

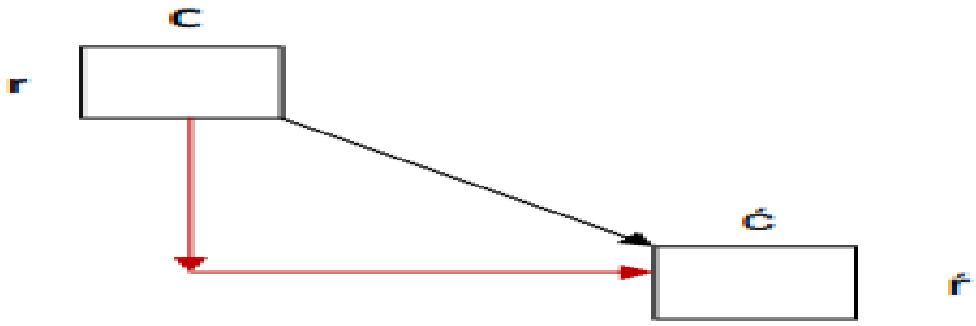
سپس در فاز سوم الگوریتم نتایج ستون سمت راست در تمامی سطرها انتشار می یابد به این ترتیب در پایان الگوریتم هر پردازنده نتایج میانی محاسباتی را نیز در خود ذخیره کرده است



Computing prefix sums on a 2D mesh

مسیر یابی packet routing برای معماری mesh دو بعدی:

به صورت دو الگوریتم بیان می شود مثلا طبق شکل بسته (r,c) می خواهد به موقعیت (r', c') برود در این صورت طبق الگوریتم first_row می تواند به سمت سطر r' حرکت کرده سپس در طول ستون ها تا ستون c' پیش برود ، یا بر عکس در الگوریتم first column می تواند به ستون c' رفته و سپس در طول سطر ها حرکت کند.



الگوریتم مرتب سازی sort مش دو بعدی:

چند تا عدد داریم قرار است مرتب سازی انجام بدهیم در دو فاز می خواهیم تعدادی عدد را در mash دو بعدی sort کنیم . الگوریتم در دو فاز انجام می شود فاز اول سطر زوج به صورت هم زمان عمل مرتب سازی را انجام می دهند و سطر های فرد را به صورت هم زمان در جهت عکس از راست به چپ انجام می دهند . تمام ستون ها به طور مستقل از بالا به پایین مرتب می شوند.

در فاز دوم تمام سطر ها از چپ به راست مرتب می شوند

تعداد تکرار فاز یکم الگوریتم از رابطه ی زیر محاسبه می شود.

فاز یک ابتدا سطر های زوج را از سمت راست به چپ مرتب می شوند ،

دوباره تکرار می کنیم به خاطر الگوریتم

فاز دو مرتب سازی سطرها از چپ به راست به صورت نزولی

الگوریتم مرتب سازی Shaer Sort برای یک معماری مش دو بعدی 2D Mesh :

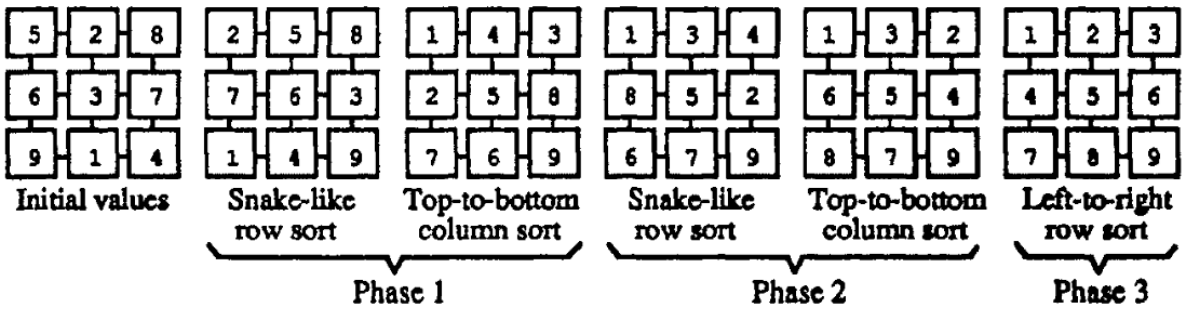
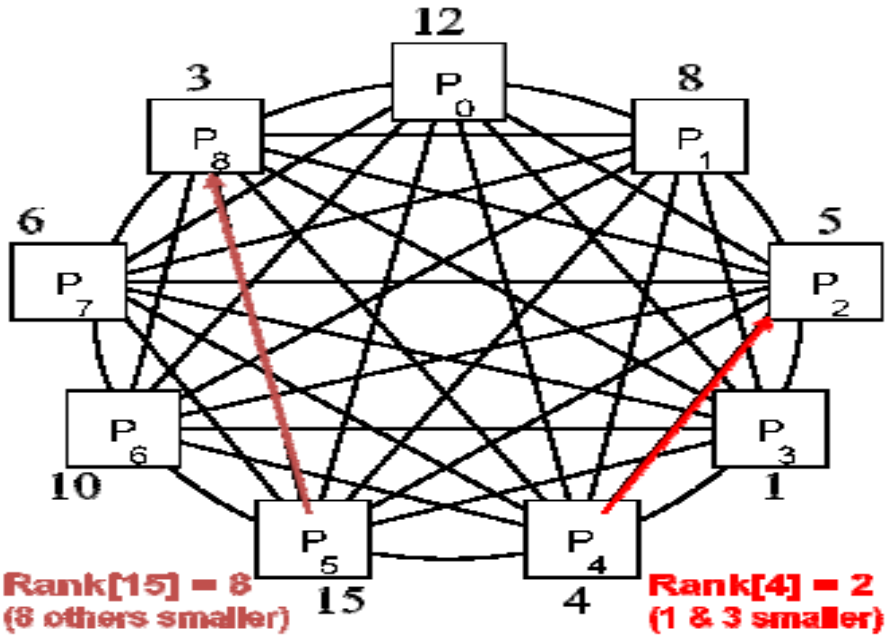


Figure 2.14. The shearsort algorithm on a 3 × 3 mesh.

جلسه ۶ پردازش موازی ۹۵/۱۰/۱

نحوه دسترسی به حافظه اشتراکی در چند پردازنده ها / نحوه دسترسی پردازنده ها به حافظه اشتراکی:



همان طور که قبلا گفته شد طبق طبقه بندی (Flynn) پردازش ها در ۴ مدل دسته بندی می شوند ،

MIMD طبقه بندی است که بیشتر مد نظر پردازش های موازیست ، این طبقه بندی مفهوم حافظه های اشتراکی را در خود دارد .

در زیر مدله‌ای که در این دسته جای میگیرن ، هر پردازنده i دریک سیکل ، سه عمل زیر را انجام می دهد (همگی از RAM) :

- ✓ **عمل واکشی: یعنی واکشی مقدار از ادرس si در حافظه اشتراکی**
- ✓ **انجام محاسبات روی داده های قرار گرفته در register های محلی**
- ✓ **ذخیره سازی یک مقدار در ادرس مقصد در حافظه اشتراکی**

نکته : در برخی از حالت ممکن است این سه مرحله به صورت کامل انجام نشود و فقط برخی از انها انجام شود .
وهمچنین ادرس های si و di در پردازنده ی pi مستقل از پردازنده های دیگر است.

انواع زیر مدلهای PRAM :

Exclusive read Exclu sive write: erew pram

وقتی که یک حافظه توسط یک پردازنده اشغال شد آن خانه منحصراً در اختیار آن پردازنده است

Exclusive read concurrent writ: ERCW PRAM

خواندن انحصاری ، نوشتن هم زمان

وقتی پردازنده ها خانه از حافظه را به منظور خواندن در اختیار گرفته ، پردازنده دیگر قادر به دسترسی به آن خانه از حافظه نمی باشند، اما می توانند همزمان خانه ای از حافظه را بنویسند (فاقد کارایی)

Exclusive read Exclu sive write: CREW PRAM

وقتی پردازنده ها خانه از حافظه را به منظور خواندن در اختیار گرفته ، پردازنده دیگر نیز قادر به خواندن از آن خانه حافظه باشند ، اما پرونده ها نمی توانند همزمان خانه ای از حافظه را بنویسند.

Concurrent read concurrent writ: CRCW PRAM

چند پردازنده می توانند همزمان از خانه ای از حافظه خوانده یا بنویسند(بسیار قوی)

انواع CRCWها:

بر اساس اینکه مدل **CRCW** نوشتن همزمان را چگونه انجام می دهد ، تعداد زیر مدلها پیشنهاد شده است .

۱. **CRCW – V (Unde firend) :** مقدار نوشته شد نا مشخص است

۲. **CRCW-D(Detecting) : تشخیص برخورد به وسیله سخت افزار :**

وقتی چند پردازنده روی یک حافظه مشترک چند مقدار متفاوت بنویسند ، سخت افزار آن را شناسایی می کند و اطلاع می دهد که برخورد به وجود آمده است.

۳. **CRCW-c (coinmon) :** اگر مقادیر نوشته شده توسط پردازنده ها مختلف یکسان باشند ، عمل نوشتن مجاز است (سازگاری)

۴. **CRCW_ R(Random) :** مقداری که قرار است دریک خانه از حافظه نوشته شود، به طور تصادفی از پردازنده ها انتخاب می شود.

۵. **CRCW-p(priority) :** پردازنده با کوچکترین شماره (یا بالعکس) یا اندیس ، عمل نوشتن را انجام می دهد .

۶. **CRCW_M(MAX_MIN) :** مقدار نوشته شده براساس بزرگترین یا کوچکترین مقدار تولید شده توسط چند پردازنده انتخاب می شود.

۷. **CRCW-S(SUM) :** مثلا روی داده های تولید شده توسط چند پردازنده که قرار است در خانه ای از حافظه ای از حافظه نوشته شوند ، عمل جمع انجام شده و حاصل جمع در حافظه ی مورد نظر نوشته می شود .

۸. **CRCW-A(AND)**

۹. **CRCW-X(XOR)**