وزارة التعليم العالي والبحث العلمي

جامعة ديالى/كلية التربية الاساسية

قسم الحاسبات

# مدخل الى الحوسبة المتوازية مفاهيم ومصطلحات

مشروع بحث مقدم الى قسم الحاسبات – كلية التربية الاساسية – جامعة ديالى

كجزء من متطلبات نيل درجة البكالوريوس تربية في الحاسبات

من قبل

سارة عبد الواحد مهدي  رانيا محمد ناصر

بأشراف

أحمد إحسان

١٤٣٩هـ

٢٠١٨م

بسم الله الرحمن الرحيم

{لا يكلف الله نفسا إلا وسعها لها ما كسبت وعليها ما اكتسبت ربنا ولا تؤاخذنا ان نسينا او أخطأنا ربنا ولا تحمل علينا إصرا كما حملته على الذين من قبلنا ربنا ولا تحملنا مالا طاقة لنا به واعف عنا واغفر لنا أنت مولانا فانصرنا على القوم الكافرين}

صدق الله العظيم

[سورة البقرة (٢٨٦)]

# الإهداء

الى الشمس التي إضاءة في سماء روحي

محمد (ص)

الى من رآني قلبها قبل ان تراني عينها

أمي الغالية

الى القلب الحنون الذي لا احد بحنانه

والدي العزيز

الى من اشد بهم أزري في الحياة

إخوتي الأعزاء

الى من أجدهم كل صباح يرتقبون نجاحي بلهفة

أساتذتي الأعزاء

إليهم جميعاً اهدي جهدي المتواضع

# الشكر والتقدير

الحمد لله حمداً كثيراً مباركاً ، يليق بجلال عظمته وعظيم سلطانه والصلاة والسلام على اشرف المرسلين وإمام النبيين سيدنا محمد الصادق الأمين معلم البشرية وعلى اله وصحبه أجمعين...

أتقدم بعظيم شكري وامتناني لأستاذي الفاضل احمد إحسان لما قدم لنا من دعم متواصل ،وزودنا بالإرشادات الدقيقة والآراء السديدة وشاركنا بجهده دون كلل او ملل طوال فترة التحضير،وأعطانا من وقته وجهده الشيء الكثير فأوجه لأستاذي بجزيل الشكر والتقدير والاحترام...

يسعدني أن أوجه شكري وتقديري لأساتذتي الأفاضل في قسم الحاسبات الذين كانوا مصدر للعطاء خلال مسيرتي الدراسية...

وواجب الوفاء والعرفان يحتم علي أن أهدي خالص الشكر والتقدير لوالدي العزيزين أمد الله في عمرهما والى إخوتي الأعزاء...

وكذلك نشكر كل من ساعد على إتمام هذا البحث وقدم لنا العون ومد يد المساعدة ولو بشيء قليل...

وأخيراً أسأل الله العلي القدير أن أكون قد وفقت في إعداد هذا البحث ومن الله العون والتوفيق...

# الفهرس

# الفصل الأول

(١-١)مقدمة البحث

(١-٢)مشكلة البحث

(١-٣)هدف البحث

(١-٤)اهمية البحث

(١-٥)الحوسبة التسلسلية (Serial Computing)

(١-٦) الحوسبة المتوازية

(١-٧) الحواسيب المتوازية (Parallel Computers)

## (١-١)مقدمة البحث

منذ ان بزغ علم الحاسبات الآلية الى الوجود والعلماء يبذلون جهدهم سعيا لجعل الحاسبات تحل المسائل بشكل افضل وأسرع،وقد أثمرة التقنية تحسنا في الدوائر الكهربائية وأصبح بالإمكان وضع العديد منها على شريحة واحدة ،كذلك ازدادت سرعة نبضة الساعة للجهاز مما أدى الى وصول سرعة المعالج الى حدود سرعات عالية تقاس بالجيجا هرتز،ومع ذلك فهناك قيود طبيعية تتحكم بالمدى الذي يمكن فيه تحسين الأداء لمعالج واحد ، فالحرارة مثلا او التشويش الكهرومغناطيسي تقللان من كثافة الترانزستورات على الشريحة، وحتى لو توصل الصناع لحل هذه المشاكل فان سرعة المعالجة لا يمكن ان تتجاوز سرعة الضوء .وعلاوة على هذه القيود الطبيعية.فثمة قيود اقتصادية،ففي وقت ما ستتزايد كلفة الإنتاج المعالج السريع جدا بشكل كبير مما قد يؤدي الى عدم الرغبة بتحمل هذه الكلفة الزائدة .كل هذه الأسباب التي ذكرناها ستؤدي في النهاية الى ترك جميع الطرق الغير مجدية وتركيز الاهتمام على طريقة واحدة وهي توزيع حمل أداء العمليات الحسابية بين عدة معالجات او ما يعرف ب"التوازي" .

## (١-٢)مشكلة البحث

تواجه المؤسسات في الوقت الحاضر العديد من المشكلات في مواكبة التغيرات في تقنيات معالجة البيانات وخصوصاً بعد ظهور أنظمة وتطبيقات عالية الإمكانات أصبح من الواجب التعريف بماهية المعالجة المتعددة او المعالجة المتوازية .

## (١-٣)هدف البحث

تقديم نظرة عامة وسريعة عن الموضوع الواسع والشامل للحوسبة المتوازية وهذا البحث يكون كمقدمة للبحوث اللاحقة،وسيغطي أساسيات الحوسبة المتوازية ،وهو موجه لمن يريد التعرف على هذا الموضوع والتعمق فيه، لعدة سنوات خلت ، كانت الحاسبات المتوازية لا توجد إلا في معامل الأبحاث . أما اليوم فهذه الحاسبات متوفرة وعلى نطاق واسع في المجالات التجارية . ان مجال المعالجة المتوازية قد نضج الى الحد الذي أصبح يدرس في المراحل الجامعية الاولى.وتجدر الإشارة الى أن التوازي مثلاً(adder) يغطي طيفا واسعا من الأشياء بداية من تصميم ابسط مكونات العتاد كالجامع وحتى تحليل النماذج النظرية للحساب المتوازي.وفي الحقيقة فان جوانب المعالجة المتوازية يمكن ان يتم دمجها في أي مقرر لعلوم الحاسب ، مثل دراسة عمارة الحاسبات ، او البرمجة،أو الشبكات او الخوارزميات وغيرها .

## (١-٤)اهمية البحث

تأتي اهمية البحث من اهمية الموضوع بحد ذاته والذي يتعلق بالحوسبة المتوازية، واستخداماتها وعرض المفاهيم والمصطلحات المرتبطة بالحوسبة المتوازية. ثم استكشاف مواضع بنيات الذاكرة المتوازية ونماذج البرمجة ولاسيما دخوله في مجال بناء الانظمة العاملة في مجال الهواتف النقالة.

## (١-٥)الحوسبة التسلسلية (Serial Computing)

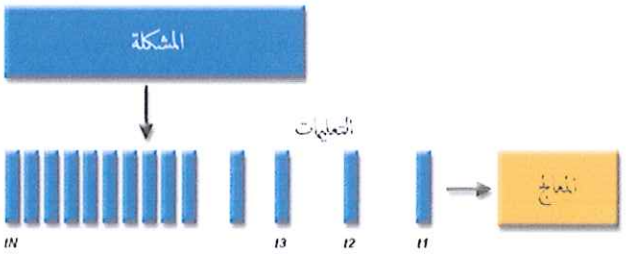
عادة تكتب البرمجيات من أجل حوسبة تسلسلية حيث:

تقسم المشكلة إلى سلسلة منفصلة من التعليمات

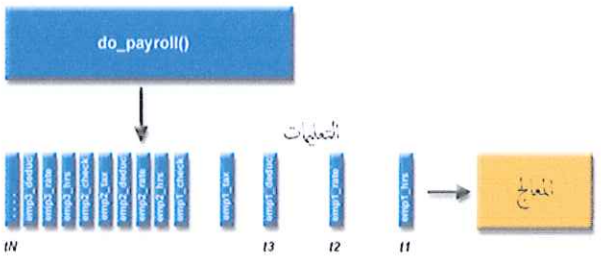
تنفذ التعليمات بالتتابع واحدة تلو الأخرى

تنفذ التعليمات على معالج واحد

يمكن تنفيذ تعليمة واحدة فقط خلال أي لحظة من الزمن



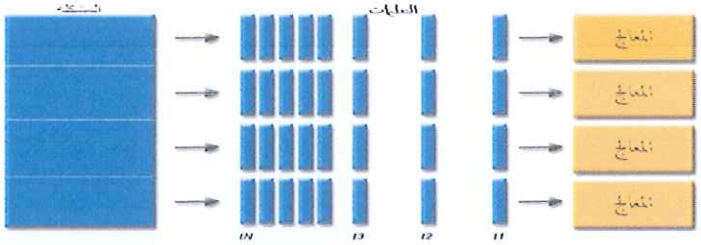

مثلا :




## (١-٦) الحوسبة المتوازية

- يُقصد بالحوسبة المتوازية، في أبسط معانيها، الاستخدام المتزامن لموارد حساب متعددة لحل مشكلة حسابية
- تقسم المشكلة إلى أجزاء منفصلة يمكن حلها في وقت واحد
- تقسم أيضا كل جزء إلى سلسلة من التعليمات

- تنفذ تعليمات كل جزء في وقت واحد على معالجات مختلفة
- تستخدم آلية شاملة للرقابة / التنسيق

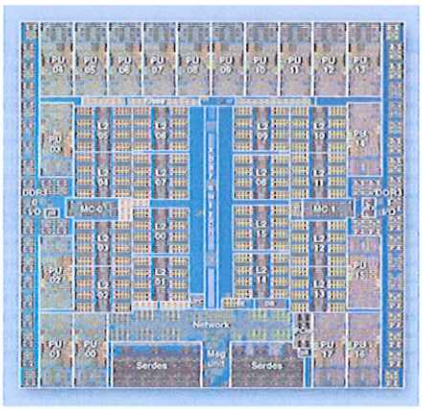


مثلا :


- يجب أن تكون المشكلة الحسابية قادرة على:
  - قابلة للتقسيم إلى قطع عمل منفصلة والتي يمكن حلها أو تنفيذها في وقت واحد؛
  - أن يتم تنفيذ تعليمات البرنامج المتعددة في أي لحظة من الزمن؛
  - أن تحل في أقل وقت مع موارد حساب متعددة بالمقارنة مع مورد حساب واحد.
- موارد الحساب عادة ما تكون:
  - جهاز حاسوب واحد مع معالجات/أنوية متعددة
  - عدد هائل من مثل هذه الحواسيب متصلة بشبكة

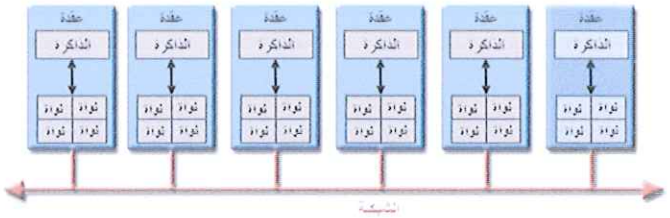
## (١-٧) الحواسيب المتوازية (Parallel Computers)

- تعتبر في يومنا هذا جميع أجهزة الحاسوب المستقلة تقريبا متوازية من منظور الأجهزة:
  - وحدات متعددة الوظائف (المخبأة أو ذاكرة التخزين المؤقت L1 cache – L1، المخبأة أو ذاكرة التخزين المؤقت L2 cache – L2 ، فرع – branch، الجلب المسبق – prefetch، فك شفرة – decode، الفاصلة العائمة – floating-point، معالجة الرسومات (GPU)، عدد صحيح – integer، وما إلى ذلك
  - وحدات/أنويه التنفيذ المتعددة
  - خيوط (threads) الأجهزة المتعددة

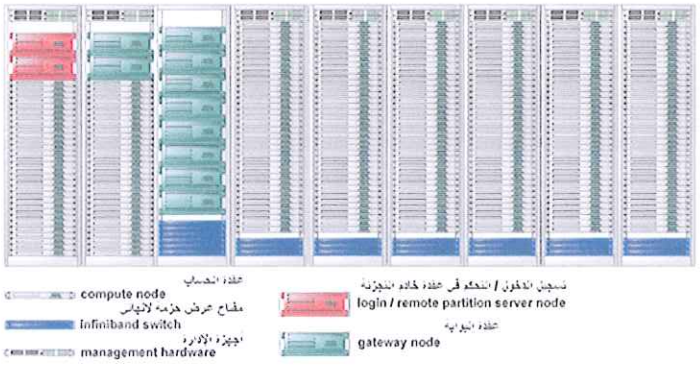


(IBM BG/Q Compute Chip with 18 cores (PU) and 16 L2 Cache units (L2)

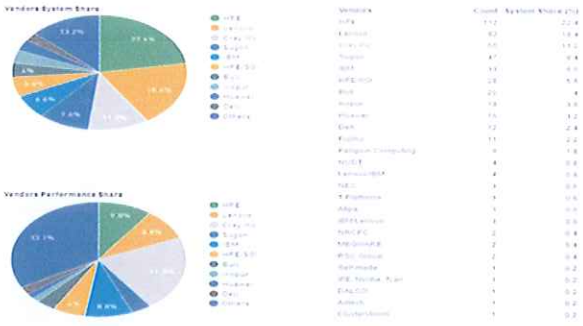
- تربط الشبكات بين العديد من الحواسب المستقلة (عقود – nodes) لإنشاء عناقيد حواسيب متوازية كبيرة



- على سبيل المثال، يظهر الرسم التخطيطي أسفله مجموعة حواسيب متوازية LLNL نموذجي:
  - كل عقدة حساب هو حاسوب متواز متعدد المعالجات في حد ذاته
  - تربط عقود حساب متعددة فيما بينها بواسطة شبكة ذات عرض حزمة لانهائي (Infiniband)
  - تستخدم العقود ذات الغرض الخاص، وكذلك متعددة المعالجات لأغراض أخرى




  - أغلبية أجهزة الحواسيب المتوازية الكبيرة العالمية (الحواسيب الفائقة– supercomputers) هي مجموعات من الأجهزة التي تنتجها حفنة من (معظم) البائعين المعروفين.



# الفصل الثاني

# مفاهيم ومصطلحات

(٢-١)المفاهيم والمصطلحات

(٢-٢)تكلفة التوازي - Parallel Overhead

(٢-٣)بنيات ذاكرة الحاسوب المتوازية

(٢-٤)تصميم البرامج المتوازية

(٢-٥)المزامنة-Synchronization

(٢-٦)الحبوبية-Granularity

(٢-٧) الإدخال والإخراج(I/O)

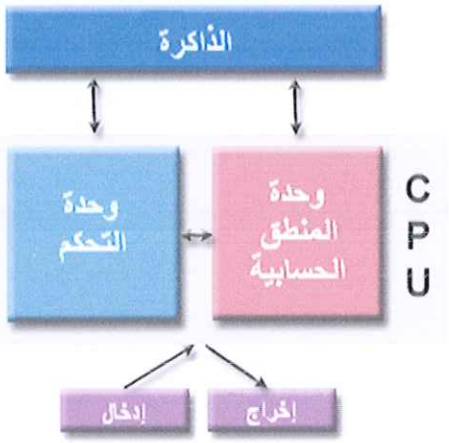
(٢-٨)التصحيح-Debugging

(٢-٩)التقنيات المستخدمة في المعالجة المتوازية

## (٢-١)المفاهيم والمصطلحات

### (٢-١-١)تصميم فون نيومان (von Neumann Architecture)

- سمي باسم عالم الرياضيات الهنغاري العبقري جون فون نيومان أول من ألف المتطلبات العامة لجهاز الحاسوب الإلكتروني في أوراقه عام ١٩٤٥.
- ويعرف أيضا باسم "حاسوب البرامج المخزَّنة (stored-program computer)" – يحتفظ بكل من تعليمات البرنامج والبيانات في الذاكرة الإلكترونية. و يختلف عن أجهزة الحاسوب السابقة التي تمت برمجتها من خلال "الأسلاك الصلبة – hard wiring".
- منذ ذلك الحين، اتبعت جميع الحواسيب تقريبا هذا التصميم الأساسي:




- تتألف من أربعة مكونات رئيسية هي:
  - الذاكرة
  - وحدة التحكم
  - وحدة المنطق الحسابية
  - الإدخال / الإخراج
- القراءة / الكتابة، تستخدم ذاكرة الوصول العشوائي لتخزين كل من تعليمات البرنامج والبيانات
- تعليمات البرنامج هي بيانات مبرمجة تخبر الحاسوب بالقيام بشيء ما
- البيانات هي ببساطة معلومات ستستخدم من قبل البرنامج
- تقوم وحدة التحكم بجلب التعليمات/البيانات من الذاكرة، و تفك شفرة التعليمات ثم تقوم بتنسيق العمليات لإنجاز المهمة المبرمجة تابعيا.
- تقوم وحدة الحساب بعمليات حسابية أساسية
- الإدخال/الإخراج (Input/output) هو واجهة المشغل البشري

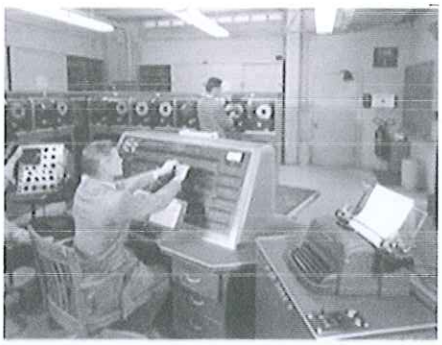
- ماذا في ذلك؟ من يهتم؟

حسنا، ما تزال أجهزة الحاسوب المتوازية تتبع هذا التصميم الأساسي، فقط ضاعفت الوحدات. بينما ظلت البنية الأساسية هي نفسها.

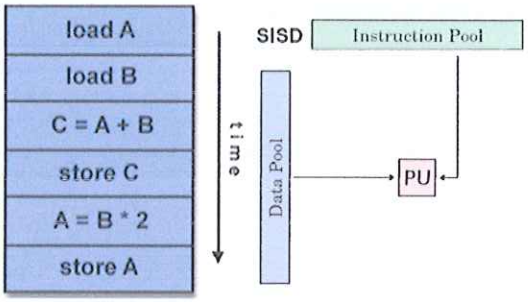
## (٢-١-٢)التصنيف الكلاسيكي لـ فلين (Flynn)

- هناك طرق مختلفة لتصنيف أجهزة الحاسوب المتوازية. الأمثلة متاحة هنا.
- واحدة من التصنيفات المستخدمة على نطاق واسع، منذ عام ١٩٦٦، تسمى تصنيف فلين.
- يميز تصنيف فلين بنيات الحاسوب متعددة المعالجات وفقا للكيفية التي يمكن أن تصنف وفقها على طول البعدين المستقلين لتدفق التعليمات (Instruction Stream) ولتدفق البيانات ( Data Stream). لكل بعد من هذه الأبعاد حالة واحدة فقط من الحالتين الممكنتين: واحدة (Single) أو متعددة (Multiple).
- تحدد المصفوفة أدناه التصنيفات الأربع الممكنة وفقا لفلين:


- **تعليمات وحيدة، بيانات وحيدة (SISD):**
  - جهاز حاسوب تسلسلي (غير متوازي)
  - تعليمات وحيدة: ينفذ تدفق تعليمات واحد فقط من قبل وحدة المعالجة المركزية (CPU) على مدار كل دورة معالجة واحدة
  - بيانات وحيدة: تستخدم تدفق بيانات وحيدة فقط كمدخلات على مدار كل دورة معالجة واحدة تنفيذ حتمي
  - هذا هو أقدم نوع من الحواسيب
  - أمثلة: أجهزة الحاسوب الكبيرة من الجيل الأقدم، أجهزة الحاسوب الصغيرة، محطات العمل وأجهزة الحاسوب الشخصية أحادية المعالج/النواة.



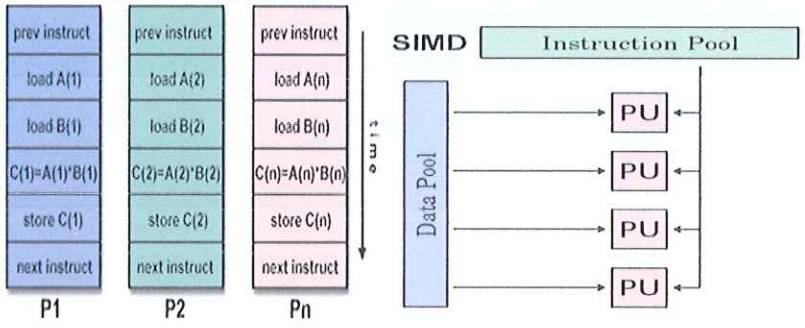
- أمثلة: أجهزة الحاسوب الكبيرة من الجيل الأقدم، أجهزة الحاسوب الصغيرة، محطات العمل وأجهزة الحاسوب الشخصية أحادية المعالج/النواة.

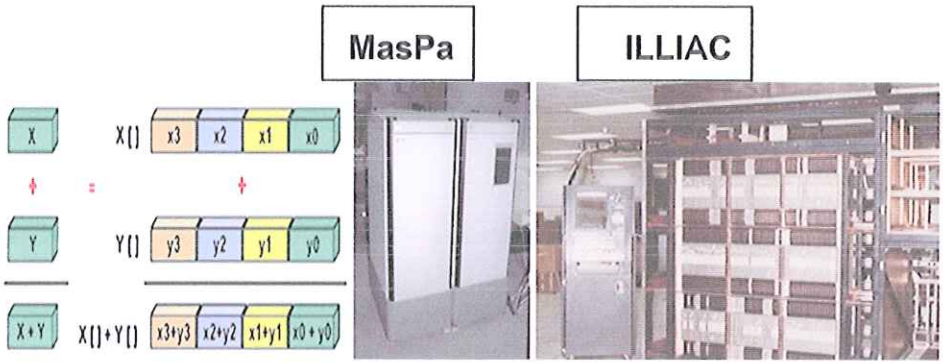




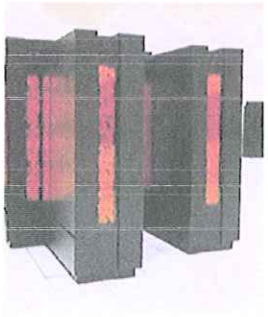

- **تعليمات وحيدة، بيانات متعددة (SIMD):**
  - نوع من أجهزة الحاسوب المتوازية
  - تعليمات وحيدة: تنفذ جميع وحدات المعالجة نفس التعليمات في أي دورة معالجة معينة
  - بيانات متعددة: يمكن لكل وحدة معالجة أن تعمل على عنصر بيانات مختلف
  - الأنسب للمشاكل المتخصصة التي تتميز بدرجة عالية من الانتظام، مثل معالجة الرسومات/الصور.
  - متزامن (بانتظام) ومنفذ حتمي
  - صنفان اثنان: مصفوفات المعالج وخطوط الأنابيب المتجهة

- أمثلة:
  - مصفوفات المعالج: MP-2, ILLIA IV & Thinking Machines CM-2, MasPar MP-1
  - خطوط الأنابيب المتجهة: C90, Fujitsu VP, NEC & IBM 9000, Cray X-MP, Y-MP SX-2, Hitachi S820, ETA10
    - معظم أجهزة الحاسوب الحديثة، وخاصة تلك التي تمتلك وحدات معالج الرسومات (GPUs) تستخدم تعليمات SIMD ووحدات التنفيذ.

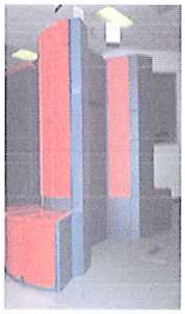



Cell Processor (GPU)



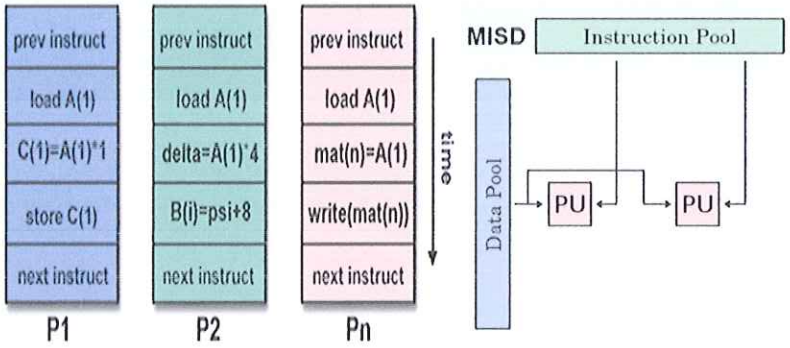
Thinking Machines CM-2



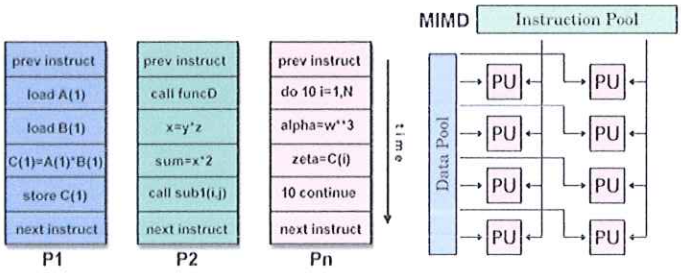
Cray Y-MP

Cray X-MP

- **تعليمات متعددة، بيانات وحيدة (MISD):**
  - نوع من أجهزة الحاسوب المتوازية
  - تعليمات متعددة: تعمل كل وحدة معالجة على البيانات بشكل مستقل عن طريق تدفقات تعليمات منفصلة.
  - بيانات وحيدة: يوزع تدفق واحد من البيانات في وحدات معالجة متعددة.
  - لا يوجد سوى عدد قليل (إن وجد) من الأمثلة الفعلية لهذه الفئة من أجهزة الحاسوب المتوازية.
  - بعض الاستخدامات التي يمكن تصورها:
  - خوارزميات تشفير متعددة تحاول كسر رسالة مشفرة واحدة.
  - مرشحات تردد متعددة تعمل على تدفق إشارة واحد




- **تعليمات متعددة، بيانات متعددة (MIMD):**
  - نوع من أجهزة الحاسوب المتوازية
  - تعليمات متعددة: قد يقوم كل معالج بتنفيذ تدفق تعليمات مختلف
  - بيانات متعددة: قد يعمل كل معالج مع تدفق بيانات مختلف
  - يمكن أن يكون التنفيذ متزامنا أو غير متزامن، حتمي أو غير حتمي
  - حاليا، هو النوع الأكثر شيوعا من أجهزة الحاسوب المتوازية – توجد معظم الحواسيب الفائقة ضمن هذه الفئة.
  - أمثلة: معظم الحواسيب الفائقة الحالية، عناقيد (clusters) و"شبكات" أجهزة حاسوب متوازية مشبكه، أجهزة حاسوب SMP متعددة المعالجات، وأجهزة حاسوب شخصية متعددة الأنوية.
  - ملاحظة: تشمل أيضا العديد من بنيات MIMD المكونات الفرعية التنفيذية SIMD



AMD Opteron

Cray XT3

IBM BG/L

## (٢-١-٣)بعض المصطلحات العامة في الحاسبات المتوازية

- مثل كل شيء آخر، الحوسبة المتوازية لها "لغة اصطلاحية" خاصة بها. وقد تم جرد بعض من المصطلحات الأكثر شيوعا والمرتبطة بالحوسبة المتوازية أسفله.
- سنتم مناقشة معظم هذه المصطلحات بمزيد من التفصيل في وقت لاحق.

### الحوسبة الفائقة / الحوسبة عالية الأداء – High Performance Computing ((HPC

تستخدم أسرع وأكبر أجهزة الحاسوب العالمية لحل المشاكل الكبيرة.

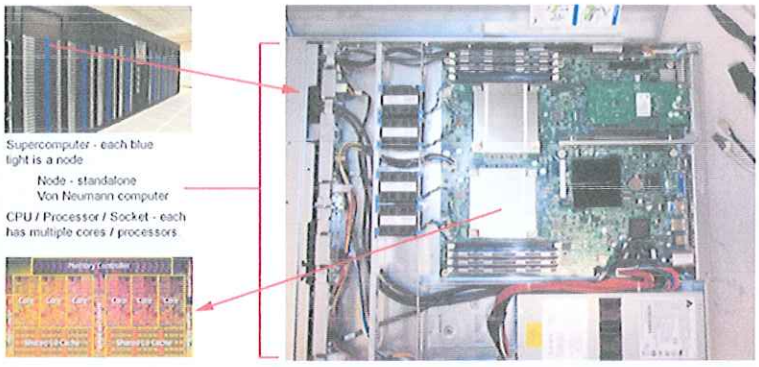
### العقدة – Node

عبارة عن "حاسوب في صندوق" مستقل من وحدات معالجة مركزية/معالجات/أنوية متعددة، ذاكرة، واجهات الشبكة، وما إلى ذلك. وتشبك العقد معا لتكون حاسوبا فائقا.

### وحدة المعالجة المركزية CPU / المقبس / المعالج / النواة – CPU / Socket / Processor / Core

هذا يختلف اعتمادا على من تتحدث إليه. في الماضي، كانت وحدة المعالجة المركزية (CPU) مكون التنفيذ الوحيد للحاسوب. ثم، دمجت وحدات معالجة مركزية متعددة في عقدة. بعد ذلك، قسمت وحدات المعالجة المركزية الفردية إلى عدة "أنوية"، كل منها أصبحت وحدة تنفيذ وحيدة. تسمى وحدات المعالجة المركزية مع الأنوية المتعددة أحيانا "مقابس" – تعتمد على المورد. والنتيجة هي عقدة ذات

وحدات معالجة مركزية متعددة، تحتوي كل منها على عدة أنوية. يخلط بين التسميات في بعض الأحيان. أتساءل لماذا؟




## المهمة – Task

قسم منفصل منطقيا من العمل الحسابي. والمهمة هي عادة برنامج أو شبه برنامج مجموعة من التعليمات التي تنفذ بواسطة معالج. ويتألف البرنامج المتوازي من مهام متعددة تشتغل على معالجات متعددة.

## الموارد – Pipelining

تقسيم مهمة إلى خطوات تؤديها وحدات المعالج المختلفة، مع مدخلات تتدفق من خلالها، يشبه بشكل كبير خط التجميع؛ نوع من الحوسبة المتوازية.

## الذاكرة المشتركة – Shared Memory

من وجهة نظر دقيقة للأجهزة، فهي تصف بنية الحاسوب حيث لدى جميع المعالجات وصول مباشر (قائم على الناقل عادة) إلى الذاكرة الملموسة المشتركة. ومن الناحية البرمجية، فإنها تصف نموذجا حيث كل المهام المتوازية لها نفس "صورة" الذاكرة ويمكنها أن تعنون وتلج مباشرة نفس مواقع الذاكرة المنطقية بغض النظر عن مكان وجود الذاكرة الملموسة فعلا.

## المعالج المتعدد المتناظر – (SMP) Symmetric Multi-Processor

بنية جهاز الذاكرة المشتركة حيث تشترك المعالجات المتعددة في مساحة عنوان واحد ولديها وصول متساو إلى جميع الموارد.

## الذاكرة الموزعة – Distributed Memory

تشير، في الأجهزة، إلى الوصول للذاكرة القائمة على الشبكة للذاكرة الملموسة غير المشتركة. وكنموذج برمجي، يمكن للمهام أن "ترى" منطقيا ذاكرة الجهاز المحلي فقط ويجب استخدام الاتصالات

للوصول إلى الذاكرة على الأجهزة الأخرى حيث يتم تنفيذ المهام الأخرى.

**الاتصالات – Communications**

عادة ما تحتاج المهام المتوازية إلى تبادل البيانات. هناك عدة طرق لتحقيق ذلك، كأن تتم عبر ناقل ذاكرة مشتركة أو عن طريق الشبكة، إلا أنه عادة ما يُشار إلى الحدث الفعلي لتبادل البيانات بالاتصالات بغض النظر عن الطريقة المستخدمة.

**التزامن – Synchronization**

هو تنسيق المهام المتوازية في الوقت الحقيقي، وغالبا ما يرتبط بالاتصالات. كما أنه في الغالب ما تنفذ عن طريق إنشاء نقطة التزامن ضمن تطبيق حيث لا يجوز لمهمة ما المضي قدما حتى تصل مهمة أو مهام أخرى إلى نفس النقطة أو ما يعادلها منطقيا. المزامنة عادة ما ينطوي على الانتظار من قبل مهمة واحدة على الأقل، وبالتالي يمكن أن يؤدي إلى زيادة في وقت تنفيذ التطبيق المتوازي.

**الحبوبية – Granularity**

في الحوسبة المتوازية، يقصد بالحبوبية ذلك المقياس النوعي للحساب بالنسبة للاتصالات.

- خشنة: يتم إجراء كميات كبيرة نسبيا من العمل الحسابي بين أحداث الاتصالات
- دقيقة: يتم إجراء كميات صغيرة نسبيا من العمل الحسابي بين أحداث الاتصالات

**التسريع المرصود – Observed Speedup**

يعرف التسريع المرصود لشفرة والذي تتم موازاته على النحو التالي:

وقت التنفيذ التسلسلي على مدار الساعة

---

وقت التنفيذ المتوازي على مدار الساعة

هو واحد من أبسط وأكثر المؤشرات استخداما وعلى نطاق واسع لأداء برنامج متوازي.

**(٢-٢)تكلفة التوازي – Parallel Overhead**

هو مقدار الوقت اللازم لتنسيق المهام المتوازية، مقابل القيام بعمل مستفاد منه. يمكن أن تشمل تكلفة التوازي عوامل مثل:

- وقت بدء المهمة
- التزامنات
- اتصالات البيانات
- تكلفة البرامج تفرضها اللغات المتوازية والمكتبات وأنظمة التشغيل وما إلى ذلك.
- وقت إنهاء المهمة

### (٢-٢-١)التوازي الضخم - Massively Parallel

يشير إلى الأجهزة التي تؤلف نظام مواز معين - تملك العديد من عناصر المعالجة. يظل معنى "العديد" يتزايد، ولكن حاليا، أكبر أجهزة الكمبيوتر المتوازية تتكون من عناصر المعالجة تحسب من مئات الآلاف إلى الملايين.

### (٢-٢-٢)التوازي المربك - Embarrassingly Parallel

حل العديد من المهام المماثلة، ولكنها مستقلة في آن واحد؛ وليس هناك حاجة إلى التنسيق بين المهام.

### (٢-٢-٣)قابلية التوسع - Scalability

تشير إلى قابلية نظام متوازي (أجهزة و/أو برمجيات) في إثبات الزيادة التناسبية في التسريع المتوازي مع إضافة المزيد من الموارد. وتشمل العوامل التي تساهم في قابلية التوسع:

- الأجهزة - خاصة حيز نطاقات ذاكرة وحدة المعالجة المركزية وخصائص اتصالات الشبكة
- خوارزمية التطبيق
- تكلفة التوازي ذات الصلة
- مميزات تطبيقك الخاص

## (٢-٣)بنيات ذاكرة الحاسوب المتوازية

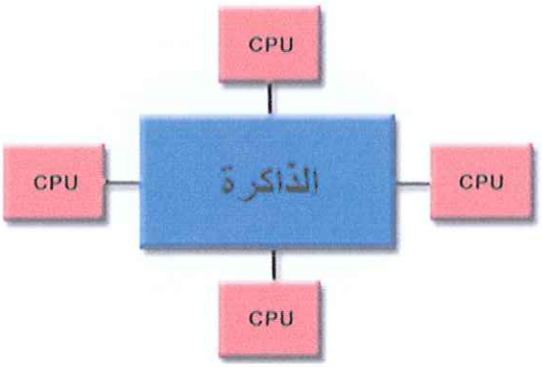
### (٢-٣-١)الذاكرة المشتركة - Shared Memory

- **الخصائص العامة**

- تختلف أجهزة الحاسوب المتوازية ذات الذاكرة المشتركة كثيرا، ولكنها تشترك عموما في قابلية جميع المعالجات في الوصول إلى كل الذاكرة كحيز عنوان عام.
- يمكن أن تعمل العديد من المعالجات بشكل مستقل ولكنها تشترك في نفس موارد الذاكرة.

- تكون التغييرات التي ينجزها معالج واحد في موقع الذاكرة مرئية لجميع المعالجات الأخرى.
- تاريخيا، تم تصنيف آلات الذاكرة المشتركة إلى ذاكرة موحدة الوصول (UMA) وذاكرة غير موحدة الوصول (NUMA)، وذلك استنادا إلى أوقات الوصول إلى الذاكرة.

- **الذاكرة موحدة الوصول – UMA) (Uniform Memory Access**

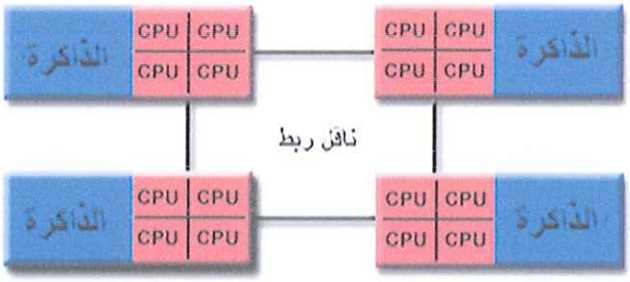



- تتمثل اليوم بشكل شائع من قبل الآلات المعالج المتعدد المتناظر (SMP)
- معالجات متطابقة
- وصول متساوي وأوقات وصول إلى الذاكرة
- تسمى أحيانا ذاكرة موحدة الوصول متماسكة مخبأة (CC-UMA). ويقصد بمتماسكة مخبأة إنه إذا قام

معالج واحد بتحديث موقع في الذاكرة المشتركة، فإن جميع المعالجات الأخرى تعلم التحديث. يتم تحقيق التماسك المخبأ على مستوى العتاد.

- الذاكرة غير موحدة الوصول – NUMA Non-Uniform) (Memory Access)

- غالبا ما تُصنع عن طريق ربط مادي (فيزيائي) لاثنين أو أكثر من الآلات المعالج المتعدد المتناظر SMP
- يمكن لآلة المعالج المتعدد المتناظر SMP الوصول مباشرة لذاكرة آلة المعالج المتعدد المتناظر SMP أخرى.
- ليس لكل المعالجات وقت وصول متساو إلى جميع الذاكرات
- يعتبر الوصول إلى الذاكرة عبر الرابط أبطأ
- إذا تم الحفاظ على التماسك المخبأ، حينها يمكن أيضا أن تسمى ذاكرة غير موحدة الوصول متماسكة مخبأة (CC-NUMA)




• الإيجابيات

• يوفر حيز العنوان العام منظور برمجة سهل الاستخدام للذاكرة

• يعد تبادل البيانات بين المهام سريعا وموحدا على حد سواء لقرب الذاكرة من وحدات المعالجة المركزية

• السلبيات

• السلبية الأساسية هي عدم قابلية التوسع بين الذاكرة ووحدات المعالجة المركزية. حيث أنه يمكن بإضافة المزيد من وحدات المعالجة المركزية أن تزيد حركة المرور هندسيا على مسار الذاكرة المشتركة لوحدة المعالجة المركزية، أما بالنسبة للأنظمة المتماسكة المخبأة، فإن حركة المرور المرتبطة بإدارة الذاكرة/المخبأة تزداد هندسيا.

• مسؤولية المبرمج لتنظيم التزامن الذي يضمن الوصول "الصحيح" للذاكرة الإجمالية.

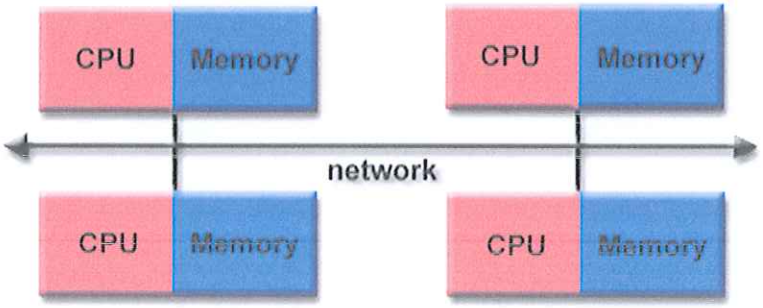
معماريات ذاكرة الحاسوب المتوازية

## (٢-٣-٢)الذاكرة الموزعة – Distributed Memory

### الخصائص العامة

مثل أنظمة الذاكرة المشتركة، تتنوع أنظمة الذاكرة الموزعة كثيرا ولكنها تتقاسم مميزات مشتركة.

وتتطلب أنظمة الذاكرة الموزعة شبكة اتصالات للتوصيل بين الذاكرة والمعالج.



- تمتلك المعالجات ذاكرة محلية خاصة بها. ولا يتم تعيين عناوين الذاكرة في معالج واحد إلى معالج آخر، لذلك ليس هناك مفهوم حيز عنوان عام عبر جميع المعالجات.
- ولأن كل معالج له ذاكرة محلية خاصة به، فإنه يعمل بشكل مستقل. إذ ليست للتغييرات التي يجريها على ذاكرته المحلية أي تأثير على ذاكرة المعالجات الأخرى. وبالتالي، فإن مفهوم التماسك المخبأ لا ينطبق.
- عندما يحتاج المعالج إلى الوصول إلى البيانات في معالج آخر، فإنه عادة ما يكون من مهمة المبرمج لتحديد صريح لكيفية وزمن توصيل البيانات. كما يعد كذلك التزامن بين المهام من مسؤولية المبرمج.
- تختلف "معمارية" الشبكة المستخدمة لنقل البيانات بشكل كبير، على الرغم من أنها يمكن أن تكون بسيطة مثل شبكة الإيثرنت.
- الإيجابيات
- تعد الذاكرة قابلة للتوسع مع عدد المعالجات. حيث كلما زاد عدد المعالجات ازداد حجم الذاكرة بشكل تناسبي.
- يمكن لكل معالج أن يصل بسرعة إلى الذاكرة الخاصة به دون تدخل ودون تكلفة المتكبدة مع محاولة الحفاظ على التماسك المخبأ العام.
- فعالية التكلفة: يمكن استخدام السلع، والمعالجات الجاهزة والتشبيك.
- السلبيات
- المبرمج هو المسؤول عن العديد من التفاصيل المرتبطة باتصال البيانات بين المعالجات.

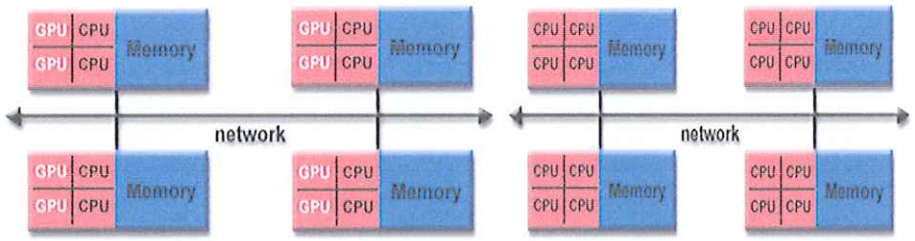
قد يكون من الصعب تعيين هياكل البيانات المتوفرة، استنادا على الذاكرة الإجمالية، إلى تنظيم الذاكرة هذا.

- أوقات الذاكرة غير موحدة الوصول – تستغرق البيانات الموجودة على عقدة بعيدة وقتا أطول للوصول مقارنة بالبيانات المحلية للعقدة

## (٢-٣-٣)الذاكرة المشتركة-الموزعة الهجينة – Hybrid Distributed-Shared Memory

### الخصائص العامة

- تستخدم اليوم أكبر وأسرع أجهزة الحاسوب في العالم معمارية الذاكرة المشتركة والموزعة على حد سواء.




- يمكن أن يكون مكون الذاكرة المشتركة جهاز ذاكرة مشترك و/أو وحدات معالجة الرسومات (GPU).
- مكون الذاكرة الموزعة عبارة عن تشبيك للعديد من أجهزة وحدات معالجة الرسومات/ذاكرات مشتركة، والتي تعرف فقط عن الذاكرة الخاصة بها – وليس عن ذاكرة على جهاز آخر. لذلك، يتطلب الأمر شبكة اتصالات لنقل البيانات من جهاز إلى آخر.
- ويبدو أن الاتجاهات الحالية تشير إلى أن هذا النوع من بنيات الذاكرة سيستمر في الانتشار وسيزداد في الحوسبة الفائقة في المستقبل المنظور.
- الإيجابيات والسلبيات
- كل شيء مشترك بين بنية كل من الذاكرة المشتركة والذاكرة الموزعة.
- زيادة قابلية التطوير هي ميزة هامة لديها
- زيادة تعقيد المبرمج هي سلبية مهم لديها

## (٢-٤)تصميم البرامج المتوازية

### (٢-٤-١)الموازاة اليدوية مقابل الموازاة التلقائية – Automatic vs. Manual Parallelization

• كان تصميم وتطوير البرامج المتوازية عملية يدوية للغاية. حيث يكون المبرمج هو المسؤول عادة عن تحديد وحاليا عن تنفيذ التوازي.

• في كثير من الأحيان، يستغرق تطوير الشفرات المتوازية يدويا وقتا طويلا، ويكون معقدا، وعرضه للخطأ وعملية تكرارية.

• منذ عدة سنوات، أصبحت أدوات مختلفة متاحة لمساعدة المبرمج على تحويل البرامج التسلسلية إلى برامج متوازية. والنوع الأكثر شيوعا من الأدوات المستخدمة لموازاة برنامج تسلسلي تلقائيا هو مترجم الموازاة أو ما قبل المعالج.

• يعمل مترجم الموازاة عموما بطريقتين مختلفتين:

تلقائية تامة – Fully Automatic

• يحلل المترجم شفرة المصدر ويحدد فرص التوازي.

• يشمل التحليل تحديد مثبطات التوازي وربما ترجيح التكلفة على ما إذا كان التوازي سيحسن الأداء فعلا أم لا.

• تعتبر حلقات التكرار (do، for) الهدف الأكثر شيوعا للتوازي التلقائي.

مبرمج موجه – Programmer Directed

• باستخدام "توجيهات المترجم" أو ربما أعلام المترجم، يقول المبرمج للمترجم بشكل واضح كيفية موازاة الشفرة البرمجية.

• قد تكون قادرة على استخدامها جنبا إلى جنب مع بضع درجات من التوازي التلقائي أيضا.

• يتم إنشاء مترجم الأكثر شيوعا للموازاة المولدة باستخدام الذاكرة المشتركة على العقدة والخيوط (مثل Open MP).

• إذا كنت تبدأ مع شفرة تسلسلية جاهزة ومقيد بوقت أو ميزانية، فإن التوازي التلقائي قد يكون هو الأفضل. ومع ذلك، هناك العديد من التحذيرات الهامة التي تنطبق على التوازي التلقائي:

◦ قد تنتج نتائج خاطئ ◦

قد يقل الأداء فعلا

◦ أقل مرونة من التوازي اليدوي

◦ تقتصر على مجموعة فرعية (معظمها حلقات التكرار) من الشفرة

◦ قد لا تكون الشفرة متوازية في الواقع إذا كان تحليل المترجم يشير إلى أن هناك مثبطات أو أن الشفرة معقدة جداً

• ينطبق الجزء المتبقي من هذا الفصل على الطريقة اليدوية لتطوير الشفرات المتوازية.

## (٢-٤-٢)فهم المشكلة والبرنامج

- مما لا شك فيه، أن الخطوة الأولى في تطوير البرمجيات المتوازية هي أولا فهم المشكلة التي ترغب في حلها بالتوازي. إذا بدأت مع برنامج تسلسلي، فإن هذا يتطلب فهم الشفرات البرمجية الجاهزة أيضا.
- قبل قضاء بعض الوقت في محاولة لتطوير حل متواز للمشكلة، حدد ما إذا كانت المشكلة هي التي يمكن أن تكون متوازية بالفعل أم لا.

○ مثال على سهولة موازاة مشكلة:

احسب الطاقة المحتملة لكل من واحدة من عدة آلاف من التشكيلات المستقلة لجزيئه. عندما تنتهي من ذلك،ابحث عن الحد الأدنى لطاقة التشكل.

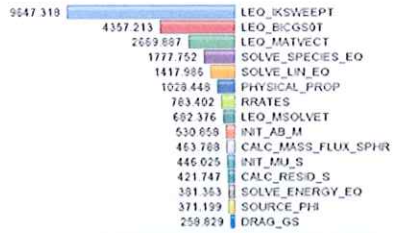
هذه المشكلة لديها القابلية على أن تحل على التوازي. حيث يمكن تحديد كل واحدة من التشكيلات الجزيئية بشكل مستقل. ويعد أيضا حساب الحد الأدنى من طاقة التشكل مشكلة متوازية.

- مثال على مشكلة مع القليل من التوازي أو بدونه:

حساب سلسلة فيبوناتشي (Fibonacci) (٠،١،١،٢،٣،٥،٨،١٣،٢١، ...) باستخدام الصيغة:

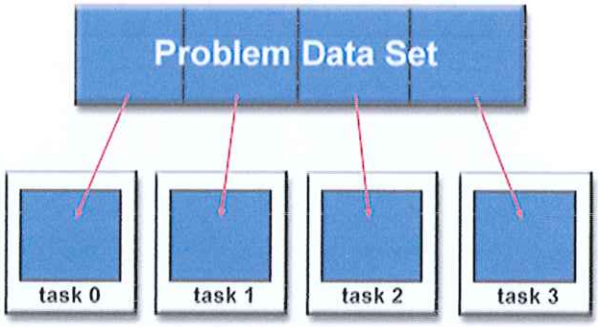
F(n) = F(n−1) + F(n−2

- يستخدم حساب القيمة F(n) قيمتي F(n−1) وF(n−2) التي يجب حسابهما أولا.

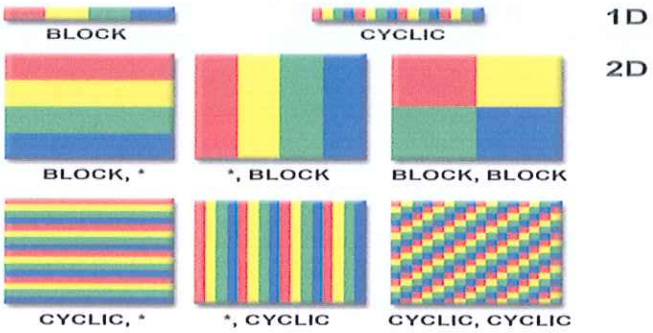



- تحديد نقاط الحوسبة المكثفة في البرنامج hotspots :
  - اعرف أين يتم معظم العمل الحقيقي. غالبا ما تحقق معظم البرامج العلمية والتقنية معظم عملها في أماكن قليلة.
  - يمكن أن تساعدك المحللات وأدوات تحليل الأداء هنا
  - ركز على موازاة نقاط الحوسبة المكثفة وتجاهل تلك الأقسام من البرنامج ذات الاستخدام القليل لوحدة المعالجة المركزية.
- تحديد نقاط الاختناق (bottlenecks) في البرنامج:
  - هل هناك مناطق بطيئة بشكل غير متناسب، أو تتسبب في وقف العمل أو إرجاءه؟ على سبيل المثال، إدخال/إخراج (I/O) هو عادة شيء يبطئ البرنامج.
  - قد يكون من الممكن إعادة هيكلة البرنامج أو استخدام خوارزمية مختلفة لتقليل أو القضاء على المناطق البطيئة غير الضرورية
- تحديد مثبطات التوازي. أحد الفئات الشائعة للمثبط هو الاعتماد على البيانات، كما يتضح من تسلسل فيبوناتشي أعلاه.
- تحقق من الخوارزميات الأخرى إن أمكن. قد يكون هذا هو الاعتبار الأكثر أهمية عند تصميم تطبيق متواز.
- استفد من إيجابيات برنامج المتوازي للطرف الثالث الأمثل ومكتبات الرياضيات المثالية المتاحة من كبار الموردين (IBM's ESSL، Intel's MKL، AMD's AMCL وما إلى ذلك).
- **(٢-٤-٣)التجزئة – Partitioning**
- واحدة من الخطوات الأولى في تصميم برنامج متواز هو تقسيم المشكلة إلى "قطع" منفصلة من الأعمال التي يمكن توزيعها على مهام متعددة. ويعرف هذا باسم التحلل أو التجزئة.
- هناك طريقتان أساسيتان لتجزئة العمل الحسابي بين المهام المتوازية: التحلل المجالي والوظيفي.
  - التحلل المجالي – Decomposition Domain:

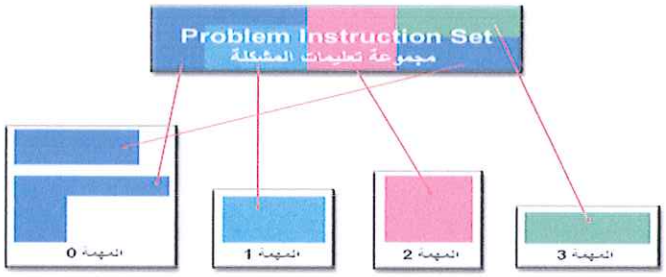
- في هذا النوع من التجزئة، يتم تحليل البيانات المرتبطة بالمشكلة. ثم تعمل كل مهمة متوازية على جزء من البيانات.




- هناك طرق مختلفة لتقسيم البيانات:




- التحلل الوظيفي – Functional Decomposition:
- في هذا النهج، ينصب التركيز على الحساب الذي يتعين القيام به وليس على البيانات التي تتم معالجتها بواسطة الحساب. وتتحلل المشكلة وفقا للعمل الذي يجب القيام به. كل مهمة تنفذ إذن جزءا من العمل الإجمالي.

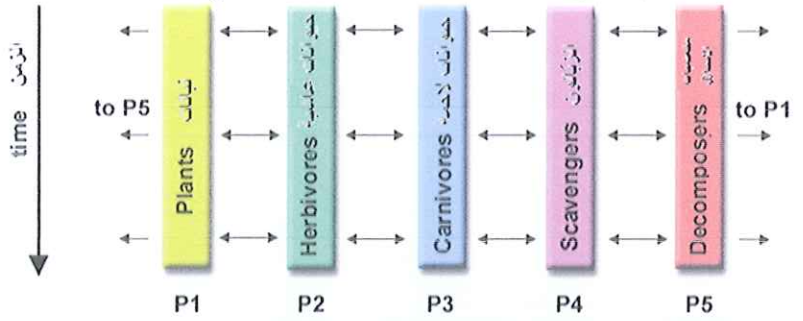



- يفسح التحلل الوظيفي المجال للمشاكل التي يمكن تقسيمها إلى مهام مختلفة. فمثلا:

نمذجة النظم الإيكولوجية – Ecosystem Modeling

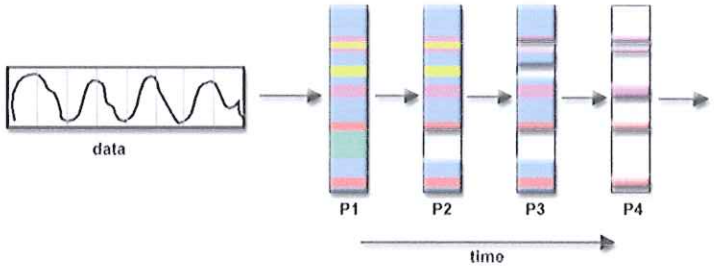
يحسب كل برنامج عدد السكان من مجموعة معينة، حيث يعتمد نمو كل مجموعة على نمو جارتها. ومع مرور الوقت، تحسب كل عملية حالتها الحالية، ثم تتبادل المعلومات مع الكثافات السكانية

المجاورة. تتقدم جميع المهام بعدها لحساب الحالة في الخطوة الزمنية التالية.



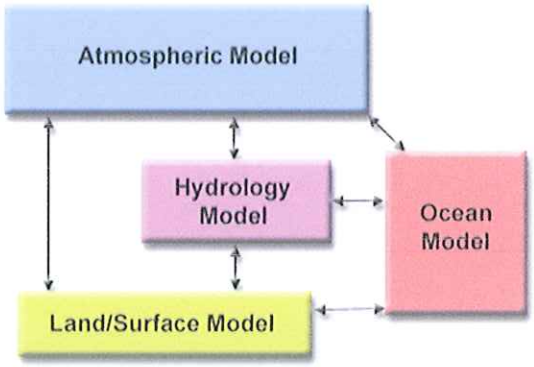

- معالجة الإشارات – Processing Signal

يتم تمرير مجموعة بيانات إشارة الصوت من خلال أربعة مرشحات حسابية مختلفة. كل مرشح عبارة عن عملية منفصلة. يجب أن يمر الجزء الأول من البيانات من خلال المرشح الأول قبل التقدم إلى الثاني. وعندما يحدث ذلك، يمر الجزء الثاني من البيانات من خلال المرشح الأول. وبحلول الوقت الذي يكون فيه الجزء الرابع من البيانات في المرشح الأول، فإن جميع المهام الأربع تكون مشغولة.




نمذجة المناخ – Modeling Climate

يمكن اعتبار كل مكون نموذج كمهمة منفصلة. وتمثل الأسهم تبادل البيانات بين المكونات أثناء الحساب: يولد نموذج الغلاف الجوي بيانات سرعة الرياح التي يستخدمها نموذج المحيطات، ونموذج المحيطات يولد بيانات درجة حرارة سطح البحر التي يستخدمها نموذج الغلاف الجوي، وهلم جرا.




- يعد الجمع بين هذين النوعين من تحلل المشكلة أمرا شائعا وطبيعيا.

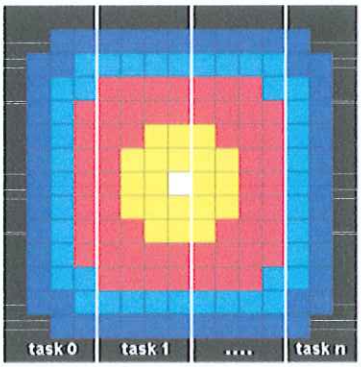
## (٢-٤-٤)الاتصالات – Communications

- من يحتاج الاتصالات؟

تتوقف الحاجة إلى الاتصالات بين المهام على مشكلتك:

تحتاج إلى الاتصالات:

- معظم التطبيقات المتوازية ليست بسيطة جدا، وتتطلب مهام لتبادل البيانات مع بعضها البعض.
- على سبيل المثال، تتطلب مشكلة انتشار الحرارة D-٢ مهمة لمعرفة درجات الحرارة التي تحسبها المهام التي لها بيانات مجاورة. للتغييرات على البيانات المجاورة تأثير مباشر على بيانات تلك المهمة.
- أنت لا تحتاج إلى الاتصالات:




- بعض أنواع المشاكل يمكن أن تتحلل وتنفذ بالتوازي دون أي حاجة تقريبا إلى المهام لتبادل البيانات.
- وغالبا ما توصف هذه الأنواع من المشاكل بالمتوازية المحرجة – وتكون هناك حاجة إلى اتصالات قليلة أو معدومة.
- على سبيل المثال، تخيل عملية معالجة الصور حيث يحتاج كل بكسل في صورة بالأبيض والأسود إلى عكس لونه. يمكن بسهولة توزيع بيانات الصورة إلى مهام متعددة تتصرف بعد ذلك بشكل مستقل عن بعضها البعض للقيام بجزئها من العمل.

- **(٢-٥)المزامنة-Synchronization**
  - إدارة تسلسل العمل والمهام التي تؤديه هو اعتبار حساس في التصميم لمعظم البرامج المتوازية.
  - يمكن أن يكون عاملا هاما في أداء البرنامج (أو عدم وجوده)
  - غالبا ما يتطلب "تسلسل" أجزاء من البرنامج.

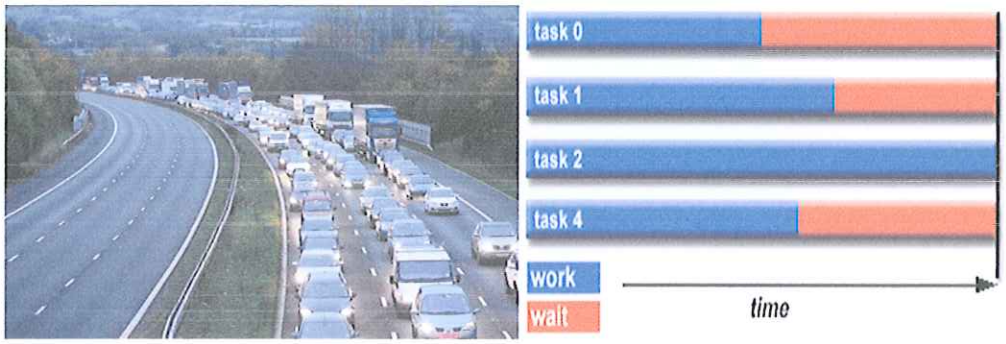
### (٢-٥-١)أنواع المزامنة:

- الحاجز – Barrier
  - عادة ما يعني أن جميع المهام معنية
  - تؤدي كل مهمة عملها حتى تصل إلى الحاجز. ثم تتوقف، أو "يتم منعها".
  - عندما تصل المهمة الأخيرة إلى الحاجز، تتم مزامنة جميع المهام.
  - ما يحدث من هنا يختلف. في كثير من الأحيان، يجب أن يتم الجزء التسلسلي من العمل. وفي حالات أخرى، يتم تحرير المهام تلقائيا لمواصلة عملها.
- قفل / سيمافور (ملوحة) – Lock / semaphore
  - يمكن أن ينطوي على أي عدد من المهام

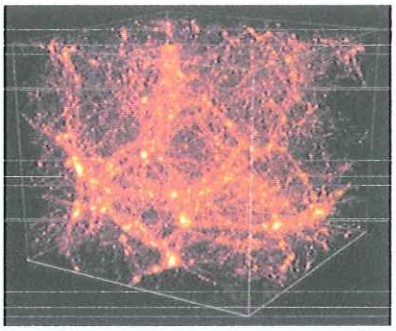
- يستخدم عادة لتسلسل (حماية) الوصول إلى البيانات الإجمالية أو جزء من الشفرة. وقد تستخدم مهمة واحدة فقط في كل مرة القفل / السيمافور / العلم (الخاص بها).
  - المهمة الأولى للحصول على قفل "يحدد" ذلك. يمكن بعدها لهذه المهمة الوصول إلى البيانات المحمية أو الشفرة بأمان (بشكل متسلسل).
  - يمكن لمهام أخرى محاولة الحصول على القفل ولكن يجب الانتظار حتى المهمة التي تملك القفل الذي يحررها.
  - يمكن أن يكون محظورا أو غير محظور

### (٢-٥-٢) موازنة التحميل - Load Balancing

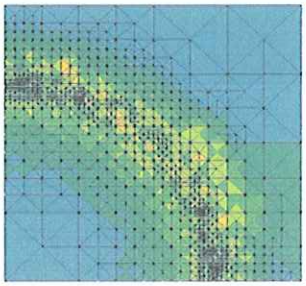
- تشير موازنة التحميل إلى ممارسة توزيع كميات متساوية تقريبا من العمل بين المهام بحيث تبقى جميع المهام مشغولة في كل وقت. ويمكن اعتباره تقليلا من وقت خمول المهمة.
- تعتبر موازنة التحميل مهمة للبرامج المتوازية لأسباب تتعلق بالأداء.على سبيل المثال، إذا كانت جميع المهام تخضع لنقطة تزامن الحاجز، فإن أبطأ مهمة تحدد الأداء العام.



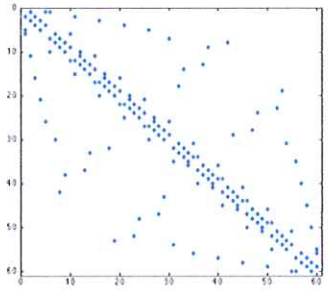

- كيفية تحقيق موازنة التحميل:
  - جزئ العمل الذي تتلقاه كل مهمة بالتساوي
    - بالنسبة لمصفوفة العمليات حيث تؤدي كل مهمة عمل مماثل، قم بتوزيع مجموعة من البيانات بين المهام بالتساوي.
    - بالنسبة لتكرار الحلقة حيث يتم إنجاز العمل في كل تكرار مماثل، قم بتوزيع التكرارات عبر المهام بالتساوي.
    - إذا تم استخدام مزيج غير متجانس من الأجهزة ذات خصائص أداء مختلفة، تأكد من استخدام بعض من أنواع أدوات تحليل الأداء للكشف عن أي اختلالات في التحميل. واضبط العمل وفقا لذلك.
  - استخدم إسناد العمل الديناميكي
    - تؤدي بعض فئات المشاكل إلى اختلالات في التحميل حتى لو كانت البيانات موزعة بالتساوي بين المهام:



محاكاة N –body – قد تهاجر الجسيمات عبر نطاقات المهام التي تتطلب المزيد من العمل لبعض المهام.

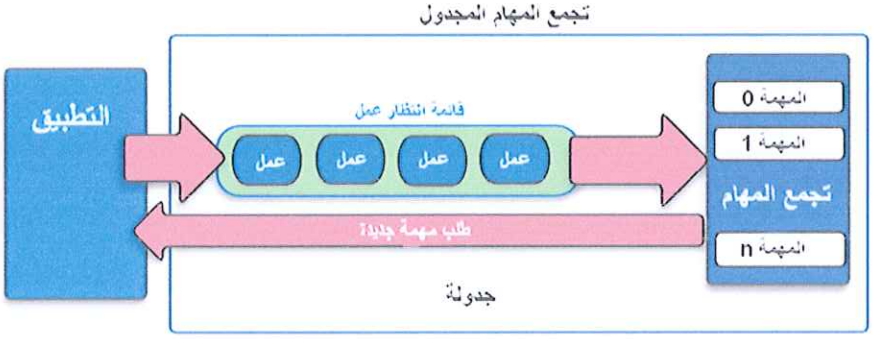


أساليب الشبكة التكيفية – قد تحتاج بعض المهام إلى تهذيب شبكتها حينما لا يقوم البعض الآخر بذلك.



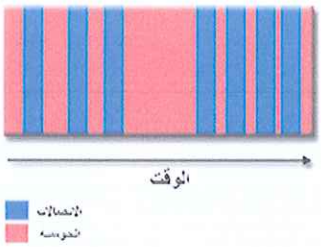
المصفوفات غير الكثيفة – بعض المهام لديها بيانات فعلية للعمل عليها حينما يكون البعض الآخر في الغالب "مجرد أصفار".

- عندما يكون مقدار العمل الذي ستؤديه كل مهمة متغيرا عن قصد أو غير قادر على التنبؤ به، فقد يكون من المفيد استخدام نهج تجمع المهام المجدول (scheduler–task pool). عندما تُنهي كل مهمة عملها، فإنها تتلقى قطعة جديدة من قائمة انتظار العمل.



◦ في نهاية المطاف، قد يصبح من الضروري تصميم خوارزمية تكشف وتتعامل مع اختلالات التحميل لأنها تحدث ديناميكيا داخل الشفرة.

## (٢-٦)الحبوبية – Granularity




• معدل الاتصالات /الحوسبة – Computation / Communication Ratio :

• في الحوسبة المتوازية، تعتبر الحبوبية مقياسا نوعيا لمعدل الحساب في الاتصالات.

• عادة ما يتم فصل فترات الحساب عن فترات الاتصالات بواسطة أحداث التزامن.

• توازي الحَب-الدقيق – Fine-grain Parallelism :

• يتم إنجاز كميات صغيرة نسبيا من العمل الحسابي بين أحداث الاتصالات

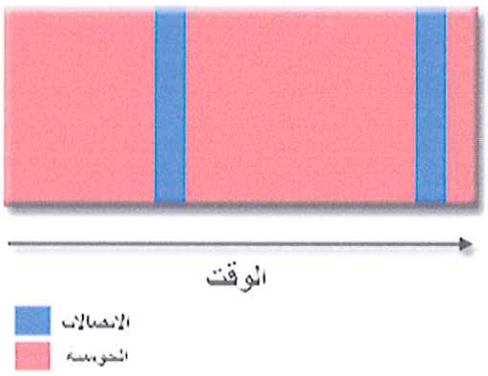
• حوسبة منخفضة إلى مقدار الاتصالات

• يسهل موازنة التحميل

• يدل على كلفة الاتصالات وفرص قليلة لتحسين الأداء

• إذا كانت الحبوبية دقيقة جدا فمن الممكن أن الكلفة المطلوبة للاتصالات والتزامن بين المهام سيستغرق وقتا أطول من الحساب.

• توازي الحَب- الخشن – Coarse-grain Parallelism :




• يتم إنجاز كميات كبيرة نسبيا من العمل الحسابي بين أحداث الاتصالات/المزامنة

• حساب مرتفع إلى مقدار نسبة الاتصالات

• يدل على المزيد من الفرص لزيادة الأداء

• تحميل الموازنة بكفاءة يكون أصعب

ما هو الأفضل؟

• تعتمد الحبوبية الأكثر كفاءة على الخوارزمية وبيئة الأجهزة التي تعمل فيها.

• في معظم الحالات تكون التكلفة المرتبطة بالاتصالات والتزامن عالية بالنسبة لسرعة التنفيذ، ولذلك فمن المفيد أن تكون لها حبوبية خشنة.

• يمكن أن يساعد توازي الحب–الدقيق على خفض الفوقانيان بسبب تحميل الموازنة.


## (٢-٧)(إدخال/إخراج)I / O

الأخبار السيئة:

•تعتبر عمليات الإدخال/الإخراج (I/O) عموما مثبطات للتوازي.

•تتطلب عمليات الإدخال/الإخراج أوامر بحجم أكبر من عمليات الذاكرة

•قد تكون أنظمة الإدخال/الإخراج المتوازية غير ناضجة أو غير متاحة لجميع الأنظمة الأساسي

• في بيئة حيث تدرك كافة المهام نفس مساحة الملف، يمكن أن تؤدي عمليات الكتابة إلى الكتابة فوق الملف.

• يمكن أن تتأثر عمليات القراءة بقدرة خادم الملفات على التعامل مع طلبات القراءة المتعددة في نفس الوقت.

• الإدخال/الإخراج الذي يجب إجراؤه عبر الشبكة (NFS، غير محلية) يمكن أن يتسبب في اختناقات شديدة وقد يسبب حتى في تعطيب خوادم الملفات.

الأخبار الجيدة:

• أنظمة الملفات المتوازية متاحة. فمثلا:

◦ GPFS: النظام العام للملفات المتوازية (IBM). يسمى حاليا مقياس طيف IBM.

◦ Lustre: لعناقيد لينكس ( Intel)

◦ HDFS نظام الملفات الموزعة Apache) Hadoop)

◦ PanFS: نظام الملفات ActiveScale Panasas لعناقيد لينكس ( Panasas، Inc.)

◦ وللمزيد – انظر

http://en.wikipedia.org/wiki/List_of_file_systems#Distributed_parallel_file_systems

- قد أصبح تخصص برمجة الإدخال/الإخراج المتوازية لـ MPI متاحة منذ عام ١٩٩٦ كجزء من MPI-2. وأصبحت عمليات تنفيذ الحرة والخاصة بالموردين متاحة بشكل شائع.
- بعض الإرشادات:
  - القاعدة رقم ١: قلل الإدخال/الإخراج الإجمالي قدر الإمكان
  - إذا كان لديك الوصول إلى نظام ملفات متوازية، استخدمه.
  - كتابة أجزاء كبيرة من البيانات بدلا من قطع صغيرة وعادة ما تكون أكثر كفاءة بكثير.
  - ملفات أقل وأكبر أداء أفضل من العديد من الملفات الصغيرة.
  - احجز الإدخال/الإخراج في أجزاء تسلسلية محددة من الوظيفة، ثم استخدم الاتصالات المتوازية لتوزيع البيانات على المهام المتوازية.على سبيل المثال، يمكن للمهمة ١ قراءة ملف إدخال ثم بعدها توصل البيانات المطلوبة إلى مهام أخرى.وبالمثل، يمكن للمهمة ١ إجراء عملية الكتابة بعد تلقي البيانات المطلوبة من جميع المهام الأخرى.
  - عمليات الإدخال/الإخراج الإجمالية عبر المهام - بدلا من تنفيذ العديد من مهام الإدخال/الإخراج، تؤدي مجموعة فرعية من المهام.

## (٢-٨)التصحيح - Debugging

- يكون من الصعب تصحيح الشفرات المتوازية بشكل لا يصدق، لا سيما الشفرات ذات المستوى الصاعد.
- الخبر السار هو أن هناك بعض المصححات الممتازة المتاحة للمساعدة:
- pthreads و Open MPمخيطة
- MPI
- GPU / مسرع
- Hybrid
- يمتلك مستخدمو حوسبة ليفرمور الوصول إلى العديد من أدوات التصحيح المتوازية المثبتة على عناقيد LC:
- Total View من Rogue Wave للبرمجيات
- DDT من Allinea
- Inspector من Intel
- أداة تحليل التتبع المكدس (STAT)- مطورة محليا

• كل هذه الأدوات لديها منحنى تعلم مرتبط بها – أكثر من غيرها.

## (٢-٩)التقنيات المستخدمة في المعالجة المتوازية

### (٢-٩-١) المعالجة باستخدام تقنية (pipelines)

و تعمل ضمن التصنيف (SISD) . يمكننا أن ننظر إلى معالجة تعليمة كسلسلة خطوات مستقلة تنفذها وحدات جزئية مستقلة من وحدة un إلى um و تستطيع هذه الوحدات الجزئية كونها مستقلة أن تتزاحم على العمل بنفس الوقت ولن يعود ضروريا أن تنتظر معالجة تعليمة ما اكتمال معالجة التعليمة السابقة لها و بالتالي في أية لحظة ستلاحظ أن وحدة التحكم تحوي عددا من التعليمات كل منها في إحدى مراحل معالجتها .

و يمكن تشبيه الوضع بوحدة تصنيع حيث تتواجد عدة وحدات من المنتج على خطوط تجميع في مواقع العمل المختلفة و كل من هذه الوحدات في إحدى مراحل تجميعها. إن الشروط الوحيدة الواجب فرضها هي :

١. يجب أن تمر كل تعليمة عبر كل المراحل أو الوحدات الجزئية ذات الصلة .

٢. تعالج لحل وحدة جزئية التعليمات دون خرق التسلسل الأصلي .

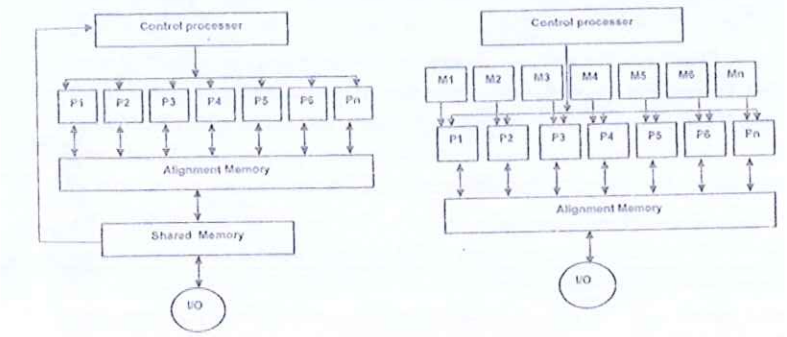
يطلق على هذه الصيغة اسم نظام أنابيب تجزئة التنفيذ حيث تعالج التعليمات كما لو أنها تنتقل عبر أنابيب متصلة و تؤدي هذه الصيغة إلى زيادة هامة في سرعة عمل الآلة . نشير هنا إلى أن الزمن الكلي اللازم لتنفيذ تعليمة واحدة يبقى كما هو إلا أن بمجموع الأزمنة التي تستغرقها محطات العمل على خط تجميع (سندعو T) زمن الحلقة الكبرى و تحتاج كل محطة إلى زمن أصغر لإتمام حصتها من عملية المعالجة و سندعو هذا الزمن t زمن الحلقة الصغرى لكل محطة . و بما أن كل محطة تنجز حصتها من عملية معالجة تعليمة واحدة في حلقة صغرى واحدة فإن أنبوب التجزئة ككل سيعالج التعليمات بمعدل تعليمة واحدة في كل حلقة صغرى ، و لولا استخدام أنابيب التجزئة لكان هذا المعدل يساوي تعليمة واحدة في كل حلقة كبرى .

## (٢-٩-٢)المعالجة باستخدام تقنية (Array Processing)

و تعمل ضمن التصنيف (SIMD) . تعرف هذه الأنظمة في بعض الأوقات بالمعالج المصفوفي حيث تتألف هذه البنى من مجموعة من المعالجات ترتبط مع بعضها على شكل مصفوفة تتألف هذه البنية من مجموعة من المعالجات المرتبطة مع بعضها و مجموعة من الذواكر المنفصلة أو تشترك بذاكرة واحدة و ميزة هذه البنية أن كل المعالجات تقوم بتطبيق نفس التعلية في نفس الوقت ولكن على معطيات مختلفة و هنا التزامن غير مطلوب بين المعالجات ،وهذا ما يبسط كثيرا تصميم هذه البنى . المعالج المركزي او المتحكم يقوم بإصدار التعليمات التي يجب تنفيذها في مصفوفة المعالجات. في الوقت الحاضر لا تتوفر مثل هذه البنى كسلع تجارية في الأسواق ، و لكن و بسبب النقص الئم الذي يتحقق في حجم الأجهزة التي نستطيع تركيبها على الرقاقات الالكترونية فإنه من المحتمل أن يكون هناك معالج مع شبكة مكوناته على رقاقة واحدة مما يجعل بنية مصفوفة المعالجات تعود فعالة اقتصاديا مرة ثانية . في الحقيقة ، غالبا ما تتشارك وحدات معالجة الإظهار (GPU) بالكثير من الخصائص مع بنى المعالج المصفوفي .

فوحدات معالجة الإظهار (Graphical Processing Units) GPU تتميز بإجراء عمليات الفاصلة العائمة (floating point) على كمية هائلة من المعطيات ، و لكي تستطيع القيام بذلك فهي تتميز ببنية داخلية مكونة من عدد كبير من المعالجات البسيطة المرتبطة مع بعضها البعض و التي تكون سريعة و لكنها تتعامل مع ذواكر محلية مرتبطة فيها (داخل بطاقة الإظهار) و هذه الذواكر محدودة ، و لكنها بالمقابل تحوي ممرات (buses) داخلية سريعة لنقل المعاملات و نتائج العمليات الحسابية التي تقوم بها . و بسبب هذه البنية (مجموعة المعالجات و الذواكر) تعتبر وحدات معالجة الإظهار من البنى التفرعية و كما تم ذكره سابقا فهي تشترك بكثير من الخصائص مع البنية –DM SIMD و من خلال هذه البنية لبطاقة الإظهار و خاصة عدد المعالجات (بشكل عام) يمكننا حل مشكلة التعليمات المتشابهة المتكررة والتي تكون معقدة بعض الاحيان على نفس المعطيات والتي غالبا ما ترد في بعض البرامج الرسومية مثل توزيع الظل على مجموعة من النقاط. اضافة الى ذلك فانه في بطاقات الاظهار الحديثة نستطيع تميز نوعين من المعالجات المختصة وذلك حسب مهاما وهي معالجات " vertex" ومعالجات "fragment" ، حيث تقوم المعالجات المصنفة "vertex" بتحديد نقاط المضلع التي سيتم رسمها و مواقعها و الوانها و تقوم المعالجات "fragment" على تعبئة الألوان و الأسطح و الإضاءة و الضلال و مع تطور المسرعات و تنوعها مثل DirectX و المميزات التي تقدمها زادت من إمكانيات ال GPU فزادت دقة عمليات الفاصلة العائمة و دخلت تقنيات حديثة مثل تقنية (HDR: High Dynamic Range) و التي فسحت المجال لتقنيات إضاءة و ظلال واقعية و كمثال على (GPU) نأخذ منتج لشركة NVIDIA والتي تعد من العمالقة في هذا المجال و هذا المنتج هو بطاقة الإظهار Tesla C1060 البنية الداخلية لهذه البطاقة تشابه

بنية المعالج المصفوفي مع وجود ذاكرة مشتركة لكل مجموعة من المعالجات البسيطة . تتميز هذه البطاقة بوجود ٢٤٠ معالج تعمل بتردد ١٠٣ غيغا هرتز و بعرض حزمة داخلي اصغر أو يساوي ١٠٢ GB/s و ذواكر ٤ غيغابايت من النوع GDDR3 إن العدد الكبير للمعالجات يجسد مبدأ تنفيذ التعليمات على التوازي و من مميزات هذه البطاقة أنها لا تحوي مخرج للفيديو و من الجدير بالذكر أن هذه البطاقات سوف تستخدم في نوع من الحواسيب الفائقة التي تعطي أداء أكثر ب ٢٥٠ مرة من الحواسيب العادية و لكن حجمه بحجم هذه الحواسيب و هذا النوع من الحواسيب غير مصمم للاستخدامات المعتادة إنما يستفاد منه في البحوث العلمية المتطورة و يعتمد أيضا على مبدأ البنى التفرعية في المعالج .

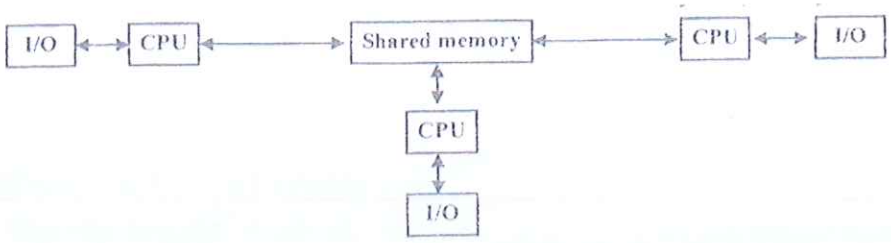



## (٢-٩-٣)المعالجة بتقنية (Multi Processer)

و تعمل ضمن التصنيف (MIMD) . يتألف هذا النظام من عدة وحدات معالجات قائمة بذاتها و التي تتصل فيما بينها بطرق معينة حسب حاجة النظام ك نحصل على زيادة في سرعة التنفيذ عن طريق المعالجة المتوازية كذلك يوفر هذا النظام وثوقية عالية كون تعطل احد وحدات المعالجة أو أكثر لا يؤدي إلى تعطل النظام بأكمله . هذه الأنظمة تتخذ ثلاثة أشكال من طرق الربط بين وحدات المعالجة المتعددة و هي :

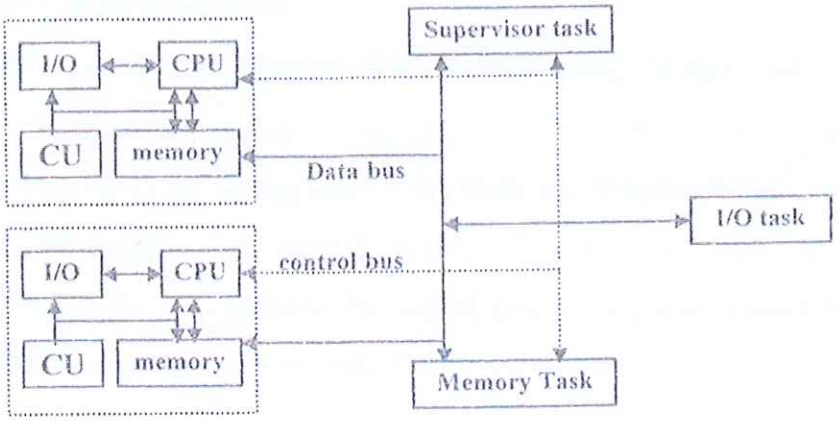
١. (Tightly Coupled System (shared memory

في هذا النوع من الربط تكون الذاكرة مركزية ترتبط بها جميع وحدات المعالجة و هذه الوحدات قادرة على تشارك العمل بنفس البرنامج أو البرامج المختلفة . و تدار هذه الوحدات من قبل نظام تشغيل واحد . و طريقة الربط موضحة في الرسم التالي :



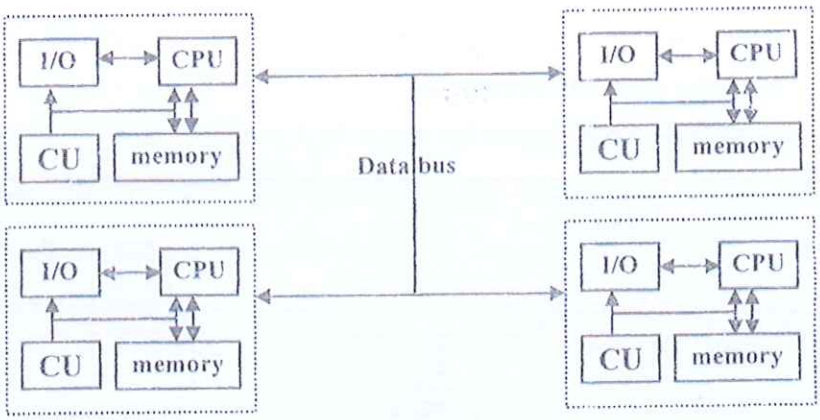

### ٢. shared task) Closely Coupled System)

في هذا النوع من الربط كل وحدة معالجة تكون متكاملة و مستقلة و قادرة على إنجاز المهام الخاصة بها و كأنها معزولة عن غيرها . و لكن عند حصول مهمة تحتاج إلى مشاركة وحدات اخرى سوف تتجه هذه المهمة إلى ذاكرة مركزية تسمى (ذاكرة المهمة المشتركة) و من ثم توزع المهام على بقية الوحدات للمعالجة و هذا ما يقوم به معالج خاص يسمى المعالج المشرف حيث في عدم وجود مشاركة في المهمات تكون هنالك لكل وحدة معالجة وحدة سيطرة خاصة بها و لكن عند حدوث المشاركة في المهام توكل السيطرة للمعالج المشرف ليكون وحدة سيطرة كاملة على جميع الوحدات . و طريقة الربط موضحة في الرسم التالي :




### ٣. shared bus) Loosely Coupled System)

في هذا النوع يكون النظام متعدد المعالجة و كل وحدة معالجة تعمل بشكل مستقل كونها نظام كامل و كل نظام يعمل بنظام تشغيل خاص به . و لكن انظمة المعالجة هذه ترتبط فيما بينها عن طريق خطوط النقل فقط . و طريقة الربط موضحة في الرسم التالي :




## (٢-٩-٤)المعالجة بتقنية (Multi Core)

مع تزايد تعقيد التطبيقات <<الملقاة على عاتق الكمبيوتر>>، حاولت الشركات المصنعة للمعالجات الاستمرار في زيادة سرعة معالجاتها للتماشي مع هذا التعقيد الذي يحتاج الى سرعة تنفيذ عالية، لكنها اصطدمت بالعديد من المعوقات اتي تتعلق بالحرارة الناتجة عن السرعات العالية جدا والزيادة في حجم المعالج بالإضافة الى زيادة التعقيد، لذا وجدت نفسها مضطرة الى البحث عن حل اخر، كتركيب معالجين مستقلين على اللوحة الاساسية، ولكن هذا الحل ايضا يترافق مع زيادة كبيرة في السعر لأنه يتطلب نوعيات خاصة من اللوحات الاساسية، لذا اتجه مهندسو الكمبيوتر الى منهجية اخرى، وهي دمج وحدتي معالجة مركزية <<CPU>> على شريحة واحدة وتخصيص موارد مستقلة لكل من هاتين النواتين، أي ان المعالج اصبح بقدرة معالجة مزدوجة ولكن مع مقبس واحد على اللوحة الاساسية، ومع سعر اقل من السعر الذي يتطلبه اقتناء معالجين مستقلين، هذا النوع من المعالجات يعرف بالمعالجات ثنائية النواة.

في حالة المعالجات وحيدة النواة، عندما يتم تشغيل عدة برامج على الكمبيوتر سيكون على المعالج تنفيذ سيل من التعليمات من تلك البرامج وسيؤدي ذلك الى زمن كبير لتنفيذ التعليمات نتيجة لحجم التعليمات الكبير، كما سيؤدي الى ضياع في الزمن لان المعالج سيحتاج الى التنقل بين التعليمات الخاصة بكل من البرامج لتنفيذها، وذلك لضمان الاستمرار في تنفيذ جميع البرامج في وقت واحد، ولكن عندما يكون المعالج ثنائي النواة تكون عملية السيطرة على سيل التعليمات اكثر سهولة وسعرة لان حاجة المعالج الى التنقل بين تعليمات البرامج المختلفة ستقل، كما ان كمية التعليمات التي يتوجب على كل وحدة معالجة تنفيذها ستصبح اقل. على الصعيد المعماري، يسمح تصميم المعالج متعدد النواة بدمج معالجين او اكثر على شريحة واحدة توضع مباشرة على مقبس واحد. لكن نظام التشغيل يرى هذه الشريحة على انها معالجين. وترتكز الفكرة الجوهرة وراء تطبيق هذه المعمارية على استراتيجية <<فرق تسد>>. بكلمات اخرى اذا استطعنا تقسيم العمليات المطلوب معالجتها على معالجين مثلا فان كل معالج سيستطيع انجاز عملياته بوقت اقل وبالتالي يستطيع المعالجين انجاز

ضعف العمل الذي يستطيعه معالج واحد. هذا الاسلوب يسمى الموازاة على مستوى المسارات<<Thread-level parallelism>> او TLP.ويستطيع المعالج المزود بتقنيةTLP تنفيذ عدة مسارات مختلفة كليا من الشيفرة، اذ يمكن ان يكون احد المسارات خاصا بتطبيق ما بينما يكون المسار الثاني مخصصا لعمليات نظام التشغيل.

ويكون الاستثمار الافضل للمعالجات ذات النوى المزدوجة من خلال نضام التشغيل ومن خلال تطبيقات الكمبيوتر نفسها، فهذه التطبيقات يجب ان تدعم تقنية <<TLP>> ليتم تنفيذها على عدة مسارات، وتحتوي جميع انظمة التشغيل المتوافرة حاليا على قطعة برمجية خاصة تسمى مجدول المهام <<Task Schedule>>، حيث يقوم مجدول المهام هذا بتوزيع التعليمات الخاصة بالبرامج العاملة على الكمبيوتر و اللازم تنفيذها كل لحظة على المعالج. وحتى ان لم يكن التطبيق يدعم المسالك المتعددة، ستكون هنالك امكانية للاستفادة من المعالجة ثنائية النواة اذا كان نظام التشغيل يدعم

<<TLP>>، فمثلا اذا كنت تستخدم نظام التشغيل ويندوز إكس بي وكان برنامج الحماية من الفيروسات يعمل في الخلفية بينما تقوم بالاستماع الى محطتك الاذاعية المفضلة عبر الانترنت، فان معالجك ثنائي النواة سينفذ التعليمات الخاصة بل برنامجين وسيعطي اداء افضل بكثير من المعالج احادي النواة. وقد اصبحت معظم البرامج في ايامنا هذه تدعم تقنية المسارات المتعددة<<multithread>> وبخاصة برامج الوسائط المتعددة مثل تلك التي تستخدم لأنشاء الافلام وانتاج الموسيقى والرسوم البيانية المتقدمة

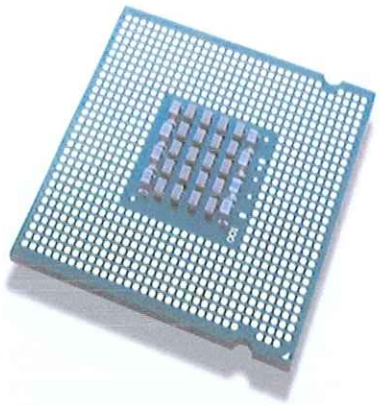
# الفصل الثالث

# تطبيقات الحوسبة المتوازية

(٣-١)المعالجات متعددة النواة

(٣-٢)مجالات استخدام الحوسبة المتوازية

(٣-٣)استعمالات الحوسبة المتوازية



## (٣-١)المعالجات متعددة النواة

يعتبر المعالج هو النواة الرئيسية لأي جهاز إلكتروني رقمي في العالم، و يكاد لا يخلو أي جهاز رقمي مهما صغر أو كبر من المعالج، فحتى الآلات الحاسبة تستخدم المعالجات لتنفيذ وظائفها الرئيسية فيمكننا تلخيص المعالج بأنه العقل المدبر لجميع أجزاء الجهاز الإلكتروني .

يتكون المعالج من عدة وحدات أساسية تتواجد في جميع المعالجات في جميع الأجهزة و تقوم بالعمليات الأساسية و هي وحدة الحساب و المنطق و المسؤولة عن العمليات الحسابية و المنطقية و المسجلات المسؤولة عن تخزين قيم و بيانات معينة و وحدة التحكم المسؤولة عن جلب التعليمات و فك تشفيرها و تنفيذهاً، و مع تطور المعالجات بدأت تظهر وحدات لا يمكن وصفها بالرئيسية لأنها تختلف باختلاف الجهاز و الشركة المصنعة، و يملك المعالج عدة أطراف تقسم لقسمين رئيسية هما أطراق العناوين و أطراف البيانات بالإضافة لأطراف أخرى لتحديد وظائف معينة، أما أطراف العناوين فهي المسؤولة عن تحديد عنوان القطعة المراد إرسال أو استقبال البيانات منها و بالتالي فإن هذه الأطراف فقط ترسل المعلومات من المعالج للجهاز لتشغيله، أما أطراف البيانات فهي المسؤولة عن إرسال البيانات من المعالج إلى القطعة التي تم تحديدها بأطراف العناوين أو استقبال البيانات منها .

بعد المقدمة البسيطة يمكننا البدء الآن بالتكلم عن المعالجات أحادية النواة و متعددة الأنوية و ما الفروق الرئيسية بينها و كيف نحدد ما هو الأنسب لجهاز معين، و لكن في البداية لنتعرف على النواة و ما الفائدة منها و وظيفتها الرئيسية.

النواة هي المعالج نفسه بجميع مكوناته و وحداته و وظائفه و بالتالي عندما نذكر النواة نحن نقصد وحدة الحساب و المنطق و وحدة التحكم و وحدة المسجلات، حيث أن هذه الوحدات الثلاثة مكملة لبعضها و تؤدي وظيفتين رئيسيتين أو طورين رئيسيين، الطور الأول هو جلب البيانات و الطور الثاني هو تنفيذ التعليمات، في الطور الأول يقوم المعالج بجلب التعليمة المراد تنفيذها و في الطور الثاني يقوم المعالج بتنفيذ التعليمة و تخزين ناتج العملية في أحد المسجلات المتخصصة و عند الانتهاء من الطور الثاني يعود للطور الأول، وهذه الفترة منذ بدايات الطور الأول و حتى نهاية الطور الثاني تسمى دورة

المعالج و تختلف التعليمات عن بعضها في عدد الدورات التي تحتاجها لتنفذ بشكل كامل فتعليمة الجمع مثلاً تحتاج لثلاثة دورات الدورة الأولى نحدد التعليمة و هي الجمع و الدورة الثانية نحدد القيمة الأولى المراد جمعها و الدورة الثالثة نحدد القيمة الثانية المراد جمعها و حين إذن نكون أتممنا عملية الجمع، و تقاس سرعة المعالجات بعدد الدورات التي يتم تنفيذها في الثانية الواحدة، فمثلاً لو أن معالج يملك سرعة ١ جيجا هيرتز هذا يعني أنه قادر على تنفيذ ١٠٠٠ دورة في الثانية الواحدة أي بمعدل ٠.٠٠١ ثانية لكل دورة. الآن و بعد أن فهمنا ما هي النواة يمكننا الانتقال و بسهولة للتحدث عن المعالجات متعددة الأنوية و ببساطة فإن المعالجات متعددة الأنوية هي تلك التي تملك أكثر من معالج داخلها، على اعتبار أن المعالج يتكون من وحدة الحساب و المنطق و وحدة التحكم و وحدة مسجلات و بالتالي فإن المعالجات ثنائية النواة مثلاً تملك وحدتان للحساب و المنطق و وحدتان تحكم و وحدتان مسجلات، و كذلك الحال للمعالجات التي تملك أكثر من نواتين، و لكن السؤال ما الفائدة من وجود معالجات متعددة الأنوية؟

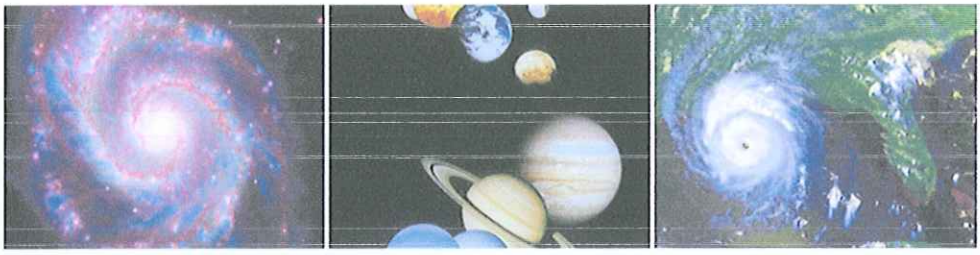
الجواب ببساطة يتعلق فقط في سرعة المعالجات، فالمعالج الذي يملك نواتين قادر على تنفيذ دورتين في نفس الوقت أي أنه أسرع بالضعف من معالج أحادي النواة بنفس التردد، فمثلاً لو أخذنا معالج بتردد ١٠٠٠ جيجا هيرتز أحادي النواة فإن الزمن لتنفيذ دورة واحدة هو ٠.٠٠١ ثانية و لكن خلال هذا الزمن سيتم تنفيذ دورة واحدة أما في المعالجات ثنائية النواة فإن الزمن لتنفيذ دورة واحدة لن يختلف و سيبقى ٠.٠٠١ ثانية و لكن سيتم تنفيذ دورتين خلال هذا الزمن و هذا لا يعني أن كل دورة تحتاج لنصف الزمن للتنفيذ، لا بل أن كل دورة ستأخذ ٠.٠٠١ ثانية و لكن لأنه يوجد نواتين فإن كل نواة تقوم بتنفيذ دورة خلال الزمن و بالتالي في نفس الزمن أحصل على دورتين، و طبعاً هذا الكلام ينطبق على المعالجات التي تحتوي على أكثر من نواتين بنفس الطريقة فالمعالج الذي يحتوي على ٤ أنوية ينفذ خلال نفس الفترة ٤ دورات .

إن القائدة الحقيقية من المعالجات متعددة الأنوية تكمن في أنه يمكن تشغيل برنامجين أو أكثر و العمل عليهم بنفس الوقت دون تأثر أحدهما بالآخر لأن كل نواة ستنفذ برنامج، و لكن بالحقيقة إن للمعالجات متعددة الأنوية استخدامات أكثر فاعلية مثل تنفيذ البرامج بطريقة أسرع و لكن كيف يتم ذلك ؟

## (٣-٢)مجالات استخدام الحوسبة المتوازية

- العالم الحقيقي متواز على نطاق واسع:
- في العالم الطبيعي، العديد من الأحداث المعقدة والمترابطة تحدث في نفس الوقت، ولكن ضمن تسلسل زمني.
- بالمقارنة مع الحوسبة التسلسلية، تعتبر الحوسبة المتوازية أكثر ملاءمة للنمذجة، ولمحاكاة وفهم، ظواهر العالم الحقيقي المعقدة.

• تخيل، على سبيل المثال، نمذجة هذه الظواهر بشكل متسلسل:



تشكل المجرة | حركات الكواكب | تغير المناخ


حركة المرور في ساعة الذروة | تكوينات الصفائح | الطقس

تجمع السيارات | صناعة طائرة | وجبة غذاء أثناء السياقة

• الأسباب الرئيسية:

• توفير الوقت و/أو المال:

◦ نظريا، يؤدي إلقاء المزيد من الموارد في مهمة ما إلى تقصير الوقت اللازم لإنجازها، مع توفير محتمل للتكاليف.

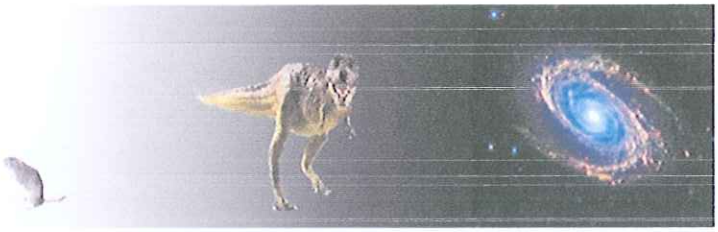
◦ يمكن تركيب الحواسيب المتوازية من مكونات السلع الأساسية الرخيصة.

• حل مشاكل معقدة إضافية/كبيرة:

◦ تعد الكثير من المشاكل جد كبيرة و/أو معقدة حيث أنه من غير العملي أو من المستحيل حلها على جهاز كمبيوتر واحد، وخاصة إذا ما نظرنا إلى ذاكرة الكمبيوتر المحدودة.

◦ مثال: "مشاكل التحدي الكبرى" (Challenge_Grand/wiki/org.wikipedia.en) التي تتطلب بيتافلوبس (PetaFLOPS) و بيتابيتس (PetaBytes) من موارد الحوسبة.

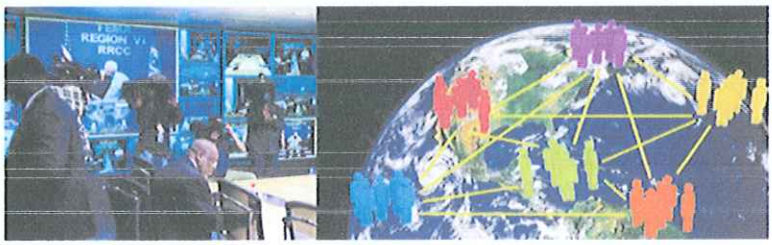
◦ مثال: تعالج محركات بحث الويب / قواعد بيانات الملايين من المعاملات في كل ثانية.



• توفير التزامن (CONCURRENCY):

◦ يمكن لمورد حساب واحد القيام بأمر واحد فقط في نفس الوقت. بينما يمكن للعديد من موارد الحساب أن تفعل أشياء كثيرة في آن واحد.

◦ مثال: توفر الشبكات التعاونية مكانا عالميا حيث يمكن للناس من جميع أنحاء العالم أن يجتمعوا ويضطلعوا بعمل "فعليا".



• الاستفادة من الموارد غير المحلية:

◦ استخدام موارد حساب على شبكة منطقة واسعة، أو حتى الإنترنت عندما تكون موارد الحوسبة المحلية نادرة أو غير كافية.

◦ مثال: تمتلك SETI@home (setiathome.berkeley.edu) أكثر من ١.٦ مليون مستخدم في كل بلد تقريبا في العالم. (يونيو ٢٠١٧).

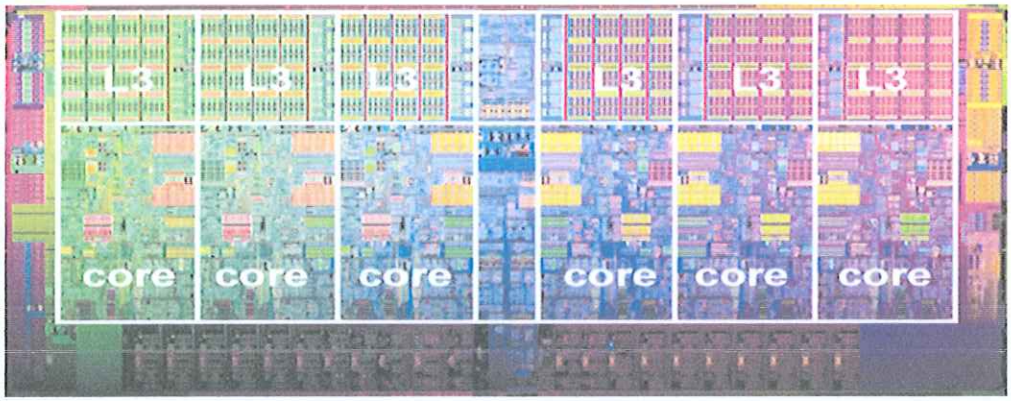
◦ مثال: تمتلك Folding@home (folding.stanford.edu) أكثر من ١.٨ مليون مساهم عالميا (يونيو ٢٠١٧)


• استخدام أفضل للأجهزة المتوازية الكامنة (UNDERLYING PARALLEL HARDWARE):

◦ تعتبر أجهزة الحاسوب الحديثة، وحتى أجهزة الحاسوب المحمول، متوازية في بنيتها مع معالجات/أنوية متعددة.

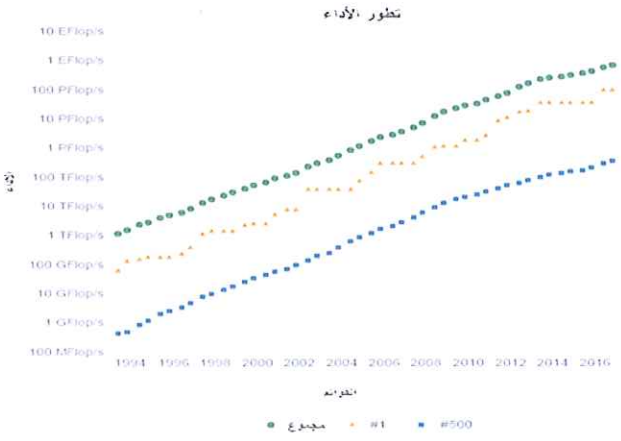
◦ يوجه البرنامج المتوازي خصيصا للأجهزة المتوازية ذات الأنوية المتعددة، الخيوط، الخ.

◦ في معظم الحالات، تعمل البرامج التسلسلية المشتغلة على أجهزة الحاسوب الحديثة على "تبديد" قدرة الحوسبة المحتملة.




Intel Xeon processor with 6 cores and 6 L3 cache units

- المستقبل
- خلال العشرين سنة الماضية، تظهر بوضوح الاتجاهات التي تشير إليها الشبكات الأسرع من أي وقت مضى، والأنظمة الموزعة، وبنيات الحواسيب متعددة المعالجات (حتى على مستوى سطح المكتب) أن التوازي هو مستقبل الحوسبة.
- في نفس هذه الفترة الزمنية، كانت هناك زيادة تفوق ٥٠٠,٠٠٠ مرة في أداء الحاسوب الفائق، مع عدم وجود نهاية في الأفق حاليا.
- السباق هو بالفعل على الحوسبة بسرعة إكساسكيل (Exascale)!
  - إكسافلوب (Exaflop) = 1018 حساب في الثانية




## (٣-٢-١)الحاجة الى استخدام التوازي

انه من المفيد الإجابة على السؤال الملح والهام :لماذا نستخدم التوازي ؟

ان السبب الرئيسي لاستعمال التوازي في تصميم البرمجيات او العتاد من اجل الحصول على الأداء الأعلى او السرعة العالية .وكل أنواع الحاسبات الضخمة اليوم تستخدم التوازي على نطاق واسع لزيادة الأداء ، فأسرع حاسب في (super computers) العالم في عام ٢٠٠٣ هو الحاسب الياباني

"محاكي الأرض" الذي يستخدم أكثر من خمسة ألاف معالج تعمل بالتوازي. ولقد زادت سرعة الحاسبات الـــى الحد الذي وصلت فيه الدارات الحاسوبية حدودا فيزيائية مثل سرعة الضوء .لذلك فمن اجل تحسين الأداء فلابد ان نستخدم التوازي

ان السرعة ليست هي السبب الوحيد في استعمال التوازي .فمصمم الحاسب يمكنه ان يضاعف من المكونات ليزيد من إمكانية الاعتماد على الجهاز الوثوقية .على سبيل المثال نظام توجيه مركبات الفضاء يتكون من ثلاثة أجهزة من الحواسيب والتي تقارن نتائج كل منهم مع الآخر ويمكن للمركبة ان تسير بجهاز واحد فقط بينما الجهازين الآخرين يكونان في وضع احتياطي.

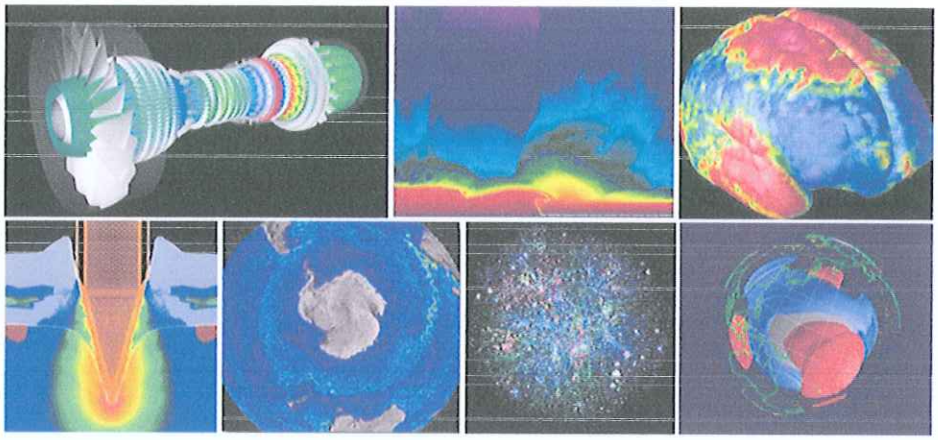
ويمكن كذلك استخدام التوازي لجعل السيطرة لامركزية . فالبنك مثلا يمكنه ان يستخدم شبكة من الحاسبات الصغيرة في المقر الرئيسي والفروع بدل من استخدام حاسب واحد كبير هذه المعالجة التقسيمية للحاسبات تتميز بوجود التحكم الداخلي عن طريق مدير البنك.

يعتبر التوازي نموذجا مهما لحل المسائل ، فالطبيعة متوازية فبينما تتحدث مع أصدقائك ،فإن القلب يضخ الدم ، والرئتين تتنفس الهواء ،والعينان تتحرك، واللسان يتحرك كل ذلك يحدث بالتوازي. والكثير من الأشياء متوازية بطبيعتها مثلا لاحظ (أفعال حشد من الناس ينتظرون المصعد للصعود الى الأعلى)أنشطة(وأفراد يقومون بضغط الزر بالطابق المتواجدون فيه)حدث (في نفس الوقت الذي يقوم أفراد مكن داخل المصعد بالضغط على الزر) ولكي نجعل الأداء أمثل باستخدام البرنامج حاسوبي مثلا(يجب التعامل مع هذه الأنشطة والأحداث المتوازية ) .

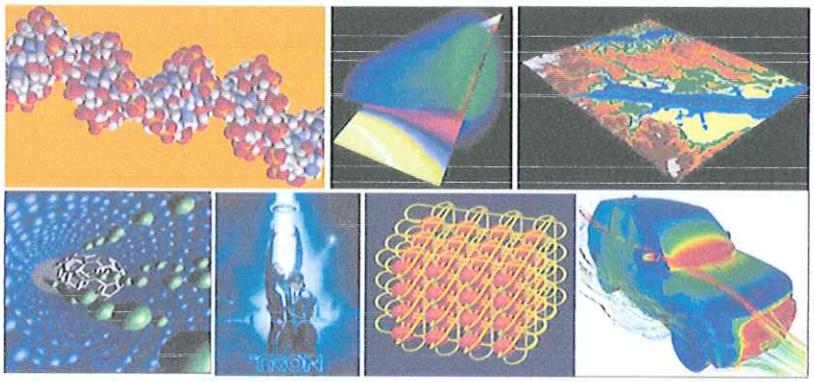
## (٣-٣)استعمالات الحوسبة المتوازية

- العلوم والهندسة
- تاريخيا، اعتبرت الحوسبة المتوازية كونها "مكرّسة للحوسبة الفائقة"، وقد استخدمت لنمذجة مشاكل صعبة في العديد من مجالات العلوم والهندسة:
- الهندسة الميكانيكية – من الأطراف الاصطناعية إلى المركبات الفضائية
- الهندسة الكهربائية، تصميم الدوائر ، الإلكترونيات الدقيقة
- علم الحاسوب، الرياضيات
- الدفاع، الأسلحة
- الغلاف الجوي، الأرض، البيئة
- الفيزياء – التطبيقية، النووية، الجسيمات، المادة المكثفة، الضغط المرتفع، الانصهار ، الضوئيات
- العلوم البيولوجية، التكنولوجيا الحيوية، علم الوراثة
- الكيمياء، العلوم الجزيئية

- الجيولوجيا، علم الزلازل

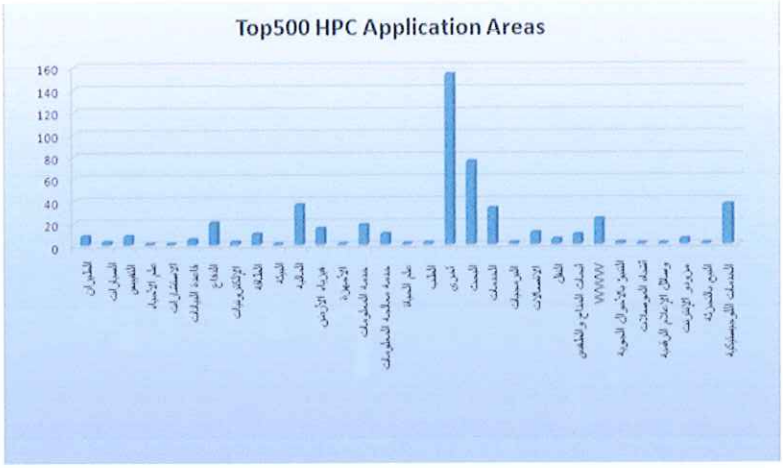


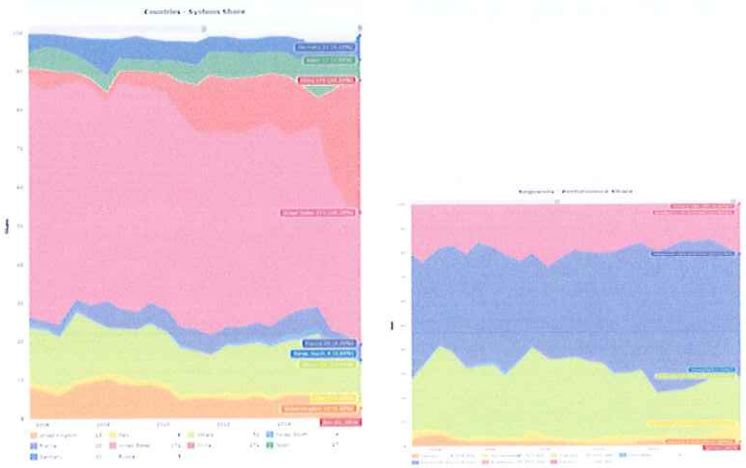
- الأغراض الصناعية والتجارية
- اليوم، توفر التطبيقات التجارية قوة دافعة متساوية أو كبيرة جدا في سبيل تطوير أجهزة الحاسوب فائقة السرعة. وتتطلب هذه التطبيقات معالجة كميات كبيرة من البيانات بطرق متطورة. مثلا:
- النمذجة المالية والاقتصادية
- إدارة الشركات الوطنية والمتعددة الجنسيات
- الرسومات المتقدمة والواقع الافتراضي، خاصة في صناعة الترفيه
- الفيديو الشبكي وتقنيات الوسائط المتعددة
- بيئات العمل التعاونية
- "البيانات الضخمة (Big Data)"، وقواعد البيانات، واستخراج البيانات
- التنقيب عن النفط
- محركات البحث على شبكة الإنترنت، خدمات الأعمال التجارية على شبكة الإنترنت
- التصوير الطبي والتشخيص
- التصميم الصيدلاني



- التطبيقات العالمية

- يجري حاليا استخدام الحوسبة المتوازية على نطاق واسع في جميع أنحاء العالم، وذلك في طائفة واسعة من التطبيقات.



المعالجة المتوازية (parallel processing):

معالجة الحاسب الالي لعدة برامج في ان واحد(سويا)وباستعمال عدة وحدات معالجة حسابية ومنطقية. وتعتبر المعالجة المتوازية حقلا جزئيا من علم الحاسب والتي تتضمن مفاهيم وافكارا من علوم الحاسب النظرية وهندسة الحاسب ولغات البرمجة والخوارزميات ومجالات التطبيق مثل الذكاء الاصطناعي والرسوم.

# الفصل الرابع

## (٤ – ١)الاستنتاجات

## (٤ – ٢)التوصيات

## (٤ – ٣)الخاتمة

## (٤-١)الاستنتاجات

ان المعالجين افضل من المعالج الواحد وذلك بسبب تقليل الجهد والوقت والمال والحوسبة المتوازية تفسر الكثير من الظواهر الطبيعية فمثلا تغير المناخ وحركة الكواكب وتشكيل المجرة تحدث في وقت واحد ولا يمكن ان تحدث واحدة بعد الأخرى، وان الحوسبة المتوازية تحل الكثير من المشاكل المعقدة فهناك مشاكل كبيرة ولا يمكن لكمبيوتر واحد حلها ، والدلالة على اهمية التوازي المتزايدة فالحاسبات الشخصية الحديثة بدأت مؤخرا بالاستفادة من التوازي بشكل عملي،فمثلا يمكن لأي شخص ان يمتلك حاسب شخصي ذا معالجين يعملان بالتوازي ومن الامثلة على المعالجة المتوازية (تعالج محركات بحث الويب /قواعد بيانات الملايين من المعاملات في كل ثانية )، وللحوسبة المتوازية استعمالات عدة منها العلوم والهندسة والرياضيات وأيضا في علوم الفضاء ، وتستعمل على نطاق واسع للإغراض الصناعية والتجارية ، اليوم توفر التطبيقات التجارية قوة دافعة متساوية او كبيرة جدا في سبيل تطوير أجهزة الحاسوب فائقة السرعة وتتطلب هذه التطبيقات معالجة كميات كبيرة من البيانات بطرق متطورة وتستعمل أيضا في التنقيب عن النفط ،ومن فوائد الحوسبة المتوازية تنفيذ المهام المستقلة بمعالجات متزامنة وبذلك تزداد نسبة العمل عند المستخدمين ،والاقلال من تكامل المعالجات في نظام وحيد الكلفة المادية لاشراكه في نظام لموارد مثل الذاكرة والاقراص ووحدات الربط مع الشبكات يقدم سرعة عالية في التوصيل بين المعالجات المتعدة وينفذ افضل تناسق واسرع وصل بين المهام المترابطة يجزئ العمل الوحيد الكبير الى عدة مهام متشابهة تنفذ في وقت واحد من اجل السرعة في التطبيق

## (٤-٢)التوصيات

١) يجب العمل على عملية التوازي للحصول على Super Computer العملاق .
٢) توسيع البحث ليشمل الجوانب العلمية و العملية لتطبيقات الحوسبة المتوازية .
٣) البحث عن مجالات تطبيق علمية اخرى لم تذكر في هذا البحث .
٤) البحث بصورة أعمق في مجالات تقنية الحوسبة المتوازية .
٥) عملية التوازي مهمة في مجالات الحاسوب لمالها من ثورة علمية كبيرة في مجال الحاسبات.
٦) وجود اكثر من معالج بحيث أن يراعى فيها عملية التوازي

## الخاتمة

يهدف هذا البحث الى تقديم نظرة سريعة عن الحوسبة المتوازية والمفاهيم والمصطلحات المتعلقة بها حيث يبدأ البحث بالتعريف بالحوسبة التسلسلية و بيان كيفية المعالجة باستخدام الحوسبة التسلسلية و من ثم يعرف الحوسبة المتوازية وما هي الحواسيب المتوازية ويبين الفرق بين الحوسبة المتوازية والحوسبة التسلسلية ثم يوضح المفاهيم والمصطلحات المتعلقة بالحوسبة المتوازية مثل تصميم فون نيومان ويبين مكونات هذا التصميم وكيفية عمله ، و التصنيف الكلاسيكي ل فلين الذي يميز بنيات الحاسوب متعددة المعالجات وفقا لطول البعدين المستقلين لتدفق التعليمات وتدفق البيانات وهذه التصنيفات هي (SISD,SIMD,MIS,MIMD) و يوضح كل تصنيف من هذه التصنيفات ثم يوضح بعض المصطلحات العامة المتوازية مثل(الحوسبة الفائقة، العقدة، وحدة المعالجة المركزية /المقبس/المعالج/النواة) ، المهمة ،الموارد ،الذاكرة المشتركة المعالج المتعدد المتناظر، الذاكرة الموزعة، الاتصالات، التزامن، الحبوبية، التسريع المرصود ) ثم يوضح ماهية تكلفة التوازي والتي تشمل(التوازي الضخم ،التوازي المربك، قابية التوسيع) ثم يتطرق البحث الى بنيات ذاكرة الحاسوب المتوازية و منها الذاكرة المشتركة ، الذاكرة الموزعة ،الذاكرة المشتركة _ الموزعة الهجينة ) ثم يبين كيفية تصميم البرامج و الموازاة اليدوية مقابل الموازاة التلقائية ثم يوضح خطوات فهم المشكلة و البرامج و كيفية تجزئة المشكلة في البرامج المتوازية ثم يبين ما هي الاتصالات و من يحتاج اليها ثم يوضح المزامنة و انواعها و ما هي موازنة التحميل و ما هي الحبوبية ثم يبين عمليات الادخال و الاخراج في الحواسيب المتوازية و يوضح كيفية تصحيح الشيفرات المتوازية و يستعرض بعض المصححات الممتازة ثم يتطرق البحث بعد ذلك الى بعض التقنيات المستخدمة في المعالجة المتوازية وهي (المعالجة باستخدام تقنية (Pipelines)،المعالجة باستخدام تقنية ( Array Processing)،المعالجة باستخدام تقنية (Multi Processer)، المعالجة بتقنية (Multi Core))، ثم يتكلم عن المعالجات متعددة النواة فيتحدث عن المعالج المتعدد النواة بالتفصيل بعد ذلك ينتقل الى مجالات استخدام الحوسبة المتوازية ثم يتطرق الى الحاجة الى استخدام التوازي و السبب الرئيسي وراء ذلك ،و يبين اهم استعمالات الحوسبة المتوازية و اخيرا يوضح فوائد تعدد المعالجات

# المراجع


- بليز بارني(Blaise Barney)،2017،مدخل الى الحوسبة المتوازية، مختبر لورانس ليفرمور الوطني

<< https://itwadi.com/Introduction_to_Parallel_Computing>>

- محمد عبد الله، بسام عبد الرحمن و آخرون، 2003، المعالجات المتوازية والخوارزميات المتوازية .

<<http://boosla.com/sh owarticle.php?sec=program&id=198 >>

- عاصم السفاريني،2012،المعالجات احادية ومتعددة النواة

<< https://www.electrony.net/57156>>

- Daniel C.Hyde. Introduction to the Principles of Parallel >>
<<Computation. Bucknell University, 1998
- Parhami, B., Introduction to Parallel Processing Algorithms and >>
<<Architectures. KLUWER ACADEMIC PUBLISHERS, 2002