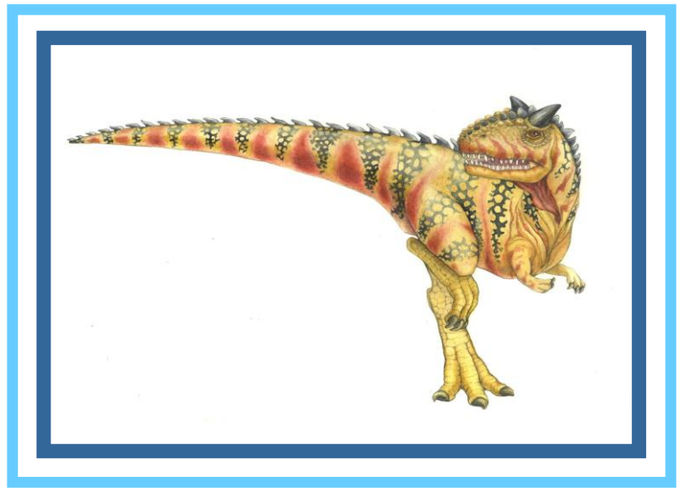
عملية الجدولة :5الفصل



Silberschatz, Galvin and Gagne ©2013

الفصل5: عملية الجدولة

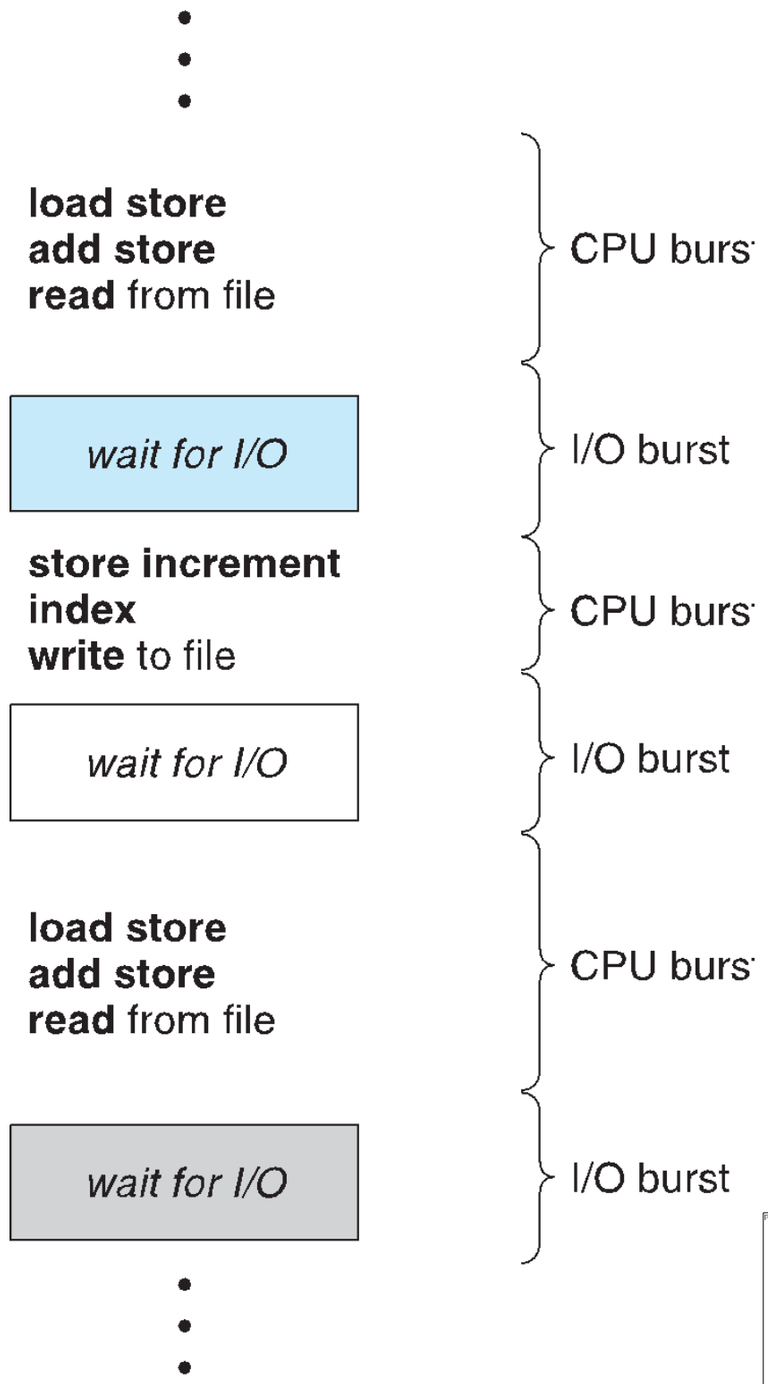
- مفاهيم أساسية
- معايير جدولة
- جدولة الخوارزميات
- جدولة الموضوع
- جدولة متعددة المعالج
- في الوقت الحقيقي وحدة المعالجة المركزية جدولة
- أمثلة نظم التشغيل
- تقييم خوارزمية

أهداف

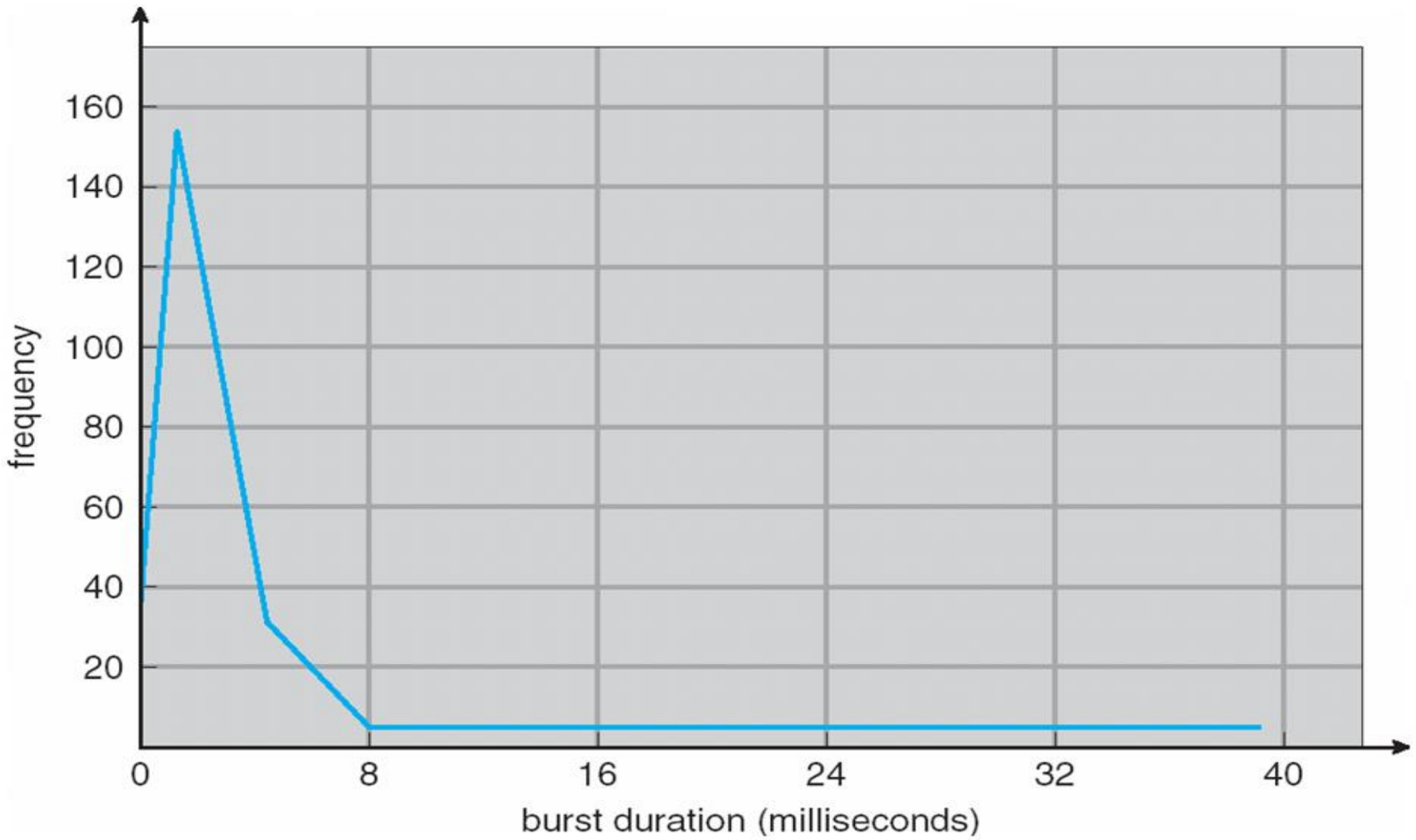
- لتقديم جدولة وحدة المعالجة المركزية، التي هي أساس لأنظمة التشغيل multiprogrammed
- لوصف مختلف الخوارزميات وحدة المعالجة المركزية جدولة
- لمناقشة معايير التقييم لاختيار خوارزمية وحدة المعالجة المركزية جدولة لنظام معين
- لدراسة خوارزميات جدولة العديد من أنظمة التشغيل

مفاهيم أساسية

- استخدام وحدة المعالجة المركزية الأقصى التي تم الحصول عليها مع متعدد البرمجة
- يتكون تنفيذ عملية ل -انفجر دورة I / O-وحدة المعالجة المركزية الانتظار I / O **دورة** من تنفيذ وحدة المعالجة المركزية و
- **انفجار I / O انفجار وحدة المعالجة المركزية** تليها
- توزيع وحدة المعالجة المركزية انفجار يثير القلق الرئيسي



الرسم البياني تايمز المتفجر وحدة المعالجة المركزية



وحدة المعالجة المركزية جدولة

- **جدولة قصيرة الأجل** يختار من بين العمليات في قائمة الانتظار على استعداد، ويخصص وحدة المعالجة المركزية إلى واحد منهم
  - طابور يجوز أن يؤمر بها بطرق مختلفة
- قرارات وحدة المعالجة المركزية جدولة قد تجري عند عملية:
  1. مفاتيح من الترشح لانتظار الدولة
  2. مفاتيح من الترشح لاستعداد الدولة
  3. مفاتيح من انتظار جاهزة
  4. ينهي
- غير **غير وقائي** 4و 1جدولة تحت
- كل جدولة الأخرى هي **استباقية**
  - النظر في الوصول إلى البيانات المشتركة
  - النظر في الشفعة في وضع النواة
  - النظر في المقاطعات التي تحدث خلال أنشطة التشغيل حاسمة

المرسل

- وهذا ينطوي .وحدة المرسل يعطي السيطرة على وحدة المعالجة المركزية لعملية اختيارهم من قبل المبرمج على المدى القصير:
  - سياق التحول
  - التحول إلى وضع المستخدم
  - القفز إلى الموقع الصحيح في برنامج المستخدم إلى إعادة تشغيل هذا البرنامج
- **إرسال الكمون** - الوقت الذي يستغرقه للمرسل لوقف عملية واحدة ويبدأ تشغيل أخرى

معايير جدولة

- الحفاظ على وحدة المعالجة المركزية مشغول ممكن -**استخدام وحدة المعالجة المركزية**
- العمليات التي تكمل تنفيذها في وحدة الزمن # -**الإنتاجية**
- كمية من الوقت لتنفيذ عملية معينة -**الفترة الزمنية**
- مقدار الوقت تم عملية الانتظار في طابور استعداد -**وقت الانتظار**
- (لاقتسام الوقت للبيئة)مقدار الوقت المستغرق من عند تقديم طلب حتى يتم إنتاج أول رد فعل، وليس الإخراج -**وقت الاستجابة**

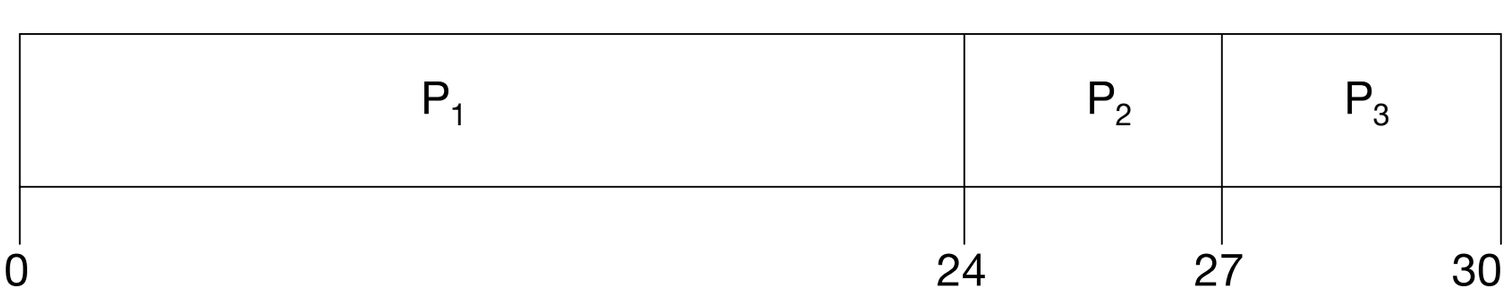
معايير تحسين جدولة خوارزمية

- استخدام وحدة المعالجة المركزية ماكس
- ماكس الإنتاجية
- دقيقة الفترة الزمنية
- دقيقة وقت الانتظار
- زمن الاستجابة دقيقة

جدولة (FCFS)تعال أولا، خدم أولا

عملية	وقت الانفجار
P_1	24
P_2	3
P_3	3

- لنفترض أن عمليات وصول بالترتيب: P_1 ، P_2 ، P_3 مخطط جانت للجدول الزمني هو:



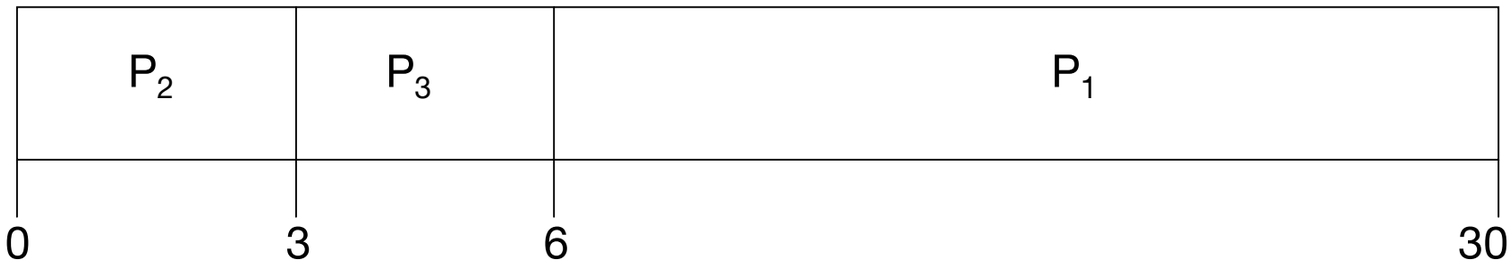
- في انتظار الوقت المناسب لP_1 = 0 ؛P_2 = 24 ؛P_3 = 27
- متوسط وقت الانتظار: (0 + 24 + 27) / 3 = 17

FCFS جدولة (يتبع)

لنفترض أن عمليات وصول بالترتيب:

P_1، P_3، P_2

n مخطط جانت للجدول الزمني هو:



n في انتظار الوقت المناسب ل $P_1 = 6$. $P_2 = 0$. $P_3 = 3$

n متوسط وقت الانتظار: (6 + 0 + 3) / 3 = 3

n أفضل بكثير من الحالة السابقة

n تأثير قافلة- عملية قصيرة وراء عملية طويلة

l تنظر أنا العمليات ملزمة- O / من والعديد المركزية المعالجة وحدة محدد واحد

جدولة (SJF)أقصر الوظيفة والعشرين

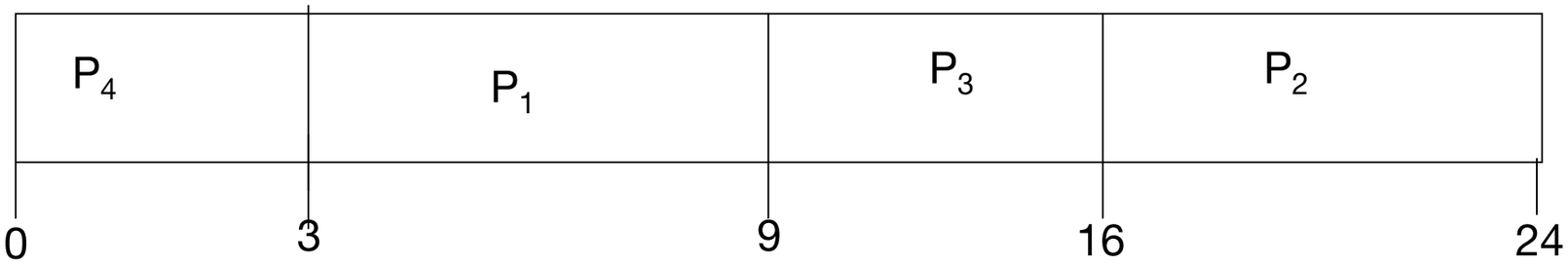
- ربط مع كل عملية على طول انفجر في وحدة المعالجة المركزية المقبل
  - استخدام هذه أطوال لجدولة عملية مع أقصر وقت

- SJF يعطي الحد الأدنى متوسط وقت الانتظار لمجموعة معينة من العمليات -هو الأمثل
  - وتكمن الصعوبة في معرفة طول طلب وحدة المعالجة المركزية المقبل
  - يمكن أن يطلب من المستخدم

مثالSJF

عملية	وقت الانفجار
P_1	6
P_2	8
P_3	7
P_4	3

- جدولة الرسم البياني SJF



- متوسط وقت الانتظار= (3 + 16 + 9 + 0) / 4 = 7

تحديد طول وحدة المعالجة المركزية التالي انفجار

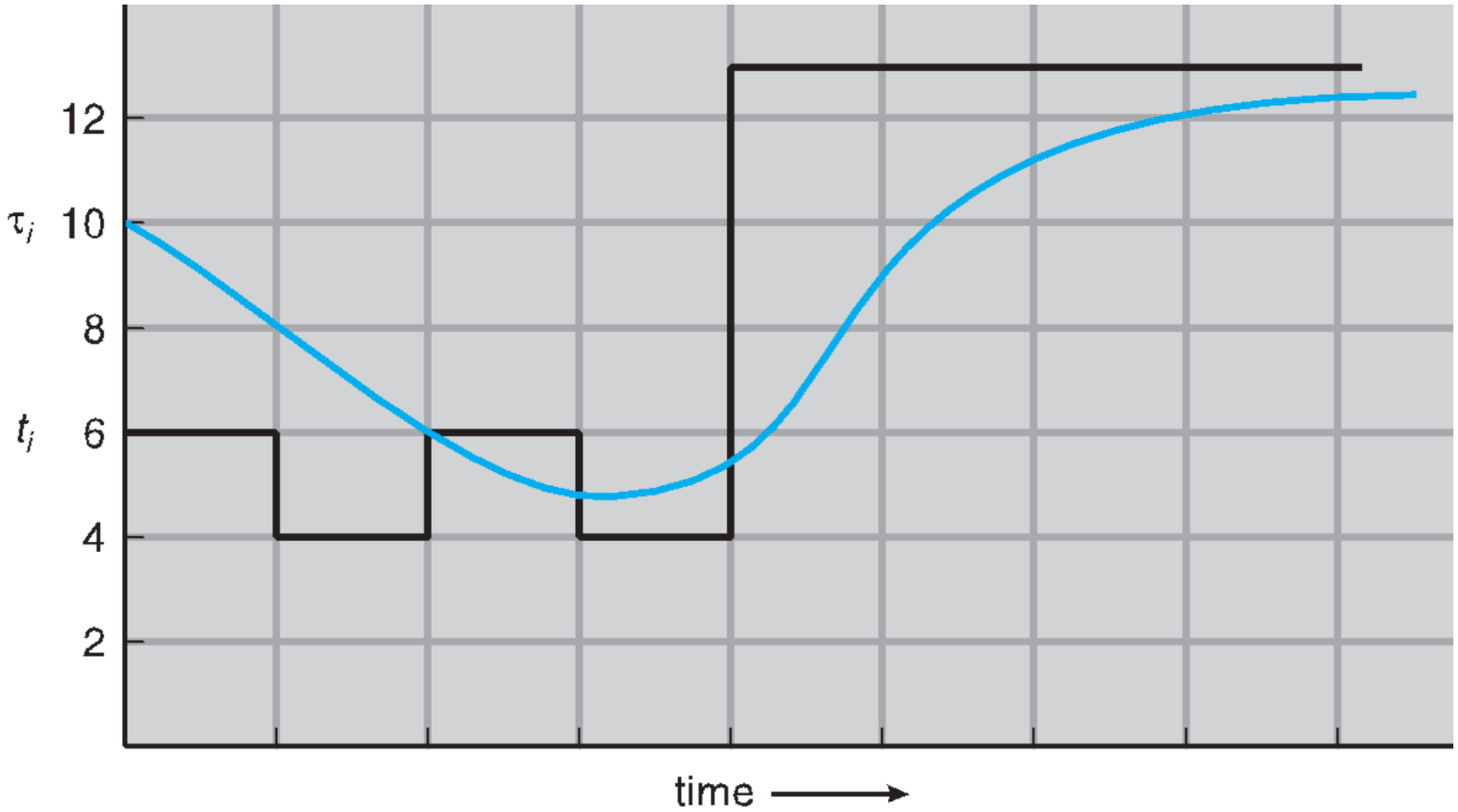
- يجب أن تكون مشابهة لسابقتها -يمكن تقدير فقط على طول
  - ثم اختيار العملية مع أقصر توقع انفجار وحدة المعالجة المركزية القادمة

- ويمكن أن يتم باستخدام طول رشقات نارية وحدة المعالجة المركزية السابقة، وذلك باستخدام المتوسط الأسي

1. t_n = actual length of n^{th} CPU burst
2. τ_{n+1} = predicted value for the next CPU burst
3. $\alpha, 0 \le \alpha \le 1$
4. Define: $\tau_{n=1} = \alpha\, t_n + (1-\alpha)\tau_n$.

- مجموعة الى نصف α عادة،
- نسخة وقائية تسمى **أقصر ما تبقى من الوقت، لأول مرة**

التنبؤ طول
وحدة المعالجة المركزية المقبل انفجار



CPU burst (t_i)	6	4	6	4	13	13	13	...
"guess" (τ_i)	10	8	6	6	5	9	11	12	...

أمثلة من المتوسط الأسي

- $\alpha = 0$
  - $\tau_{n+1} = \tau_n$
  - التاريخ الحديث لا تعول
- $\alpha = 1$
  - $\tau_{n+1} = \alpha t_n$
  - فقط الفعلية وحدة المعالجة المركزية مشاركة انفجار التهم
- إذا كان لنا أن توسيع الصيغة، نحصل على:

$$\tau_{n+1} = \alpha t_n + (1 - \alpha)\alpha t_{n-1} + \ldots + (1 - \alpha)^j \alpha t_{n-j} + \ldots + (1 - \alpha)^{n+1} \tau_0$$

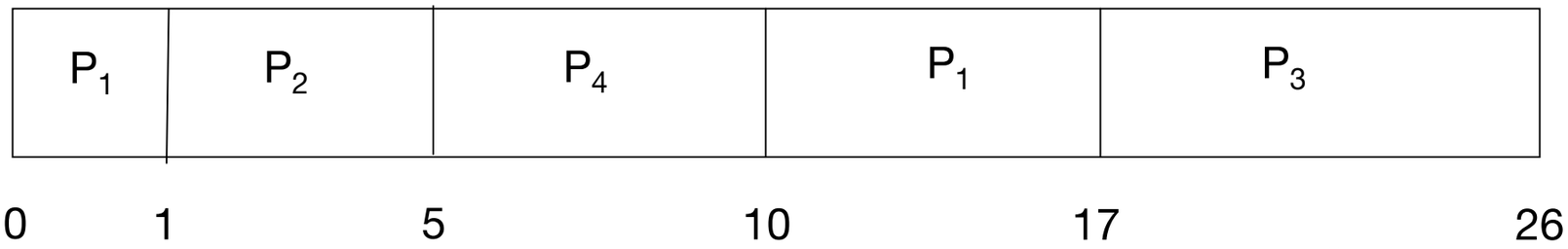
- ، كل فترة لاحقة له وزن أقل من سابقتها1هي أقل من أو يساوي $(1 - \alpha)$و αونظرا لأن كلا

مثال أقصر المتبقية من الوقت، لأول مرة

- الآن نضيف مفاهيم أوقات وصول والاستباق مختلفة لتحليل

عملية	*وصول* مرة	وقت الانفجار
P_1	0	8
P_2	1	4
P_3	2	9
P_4	3	5

- مخطط جانت SJF *وقائي*



- متوسط وقت الانتظار = [(1/10) + (1/1) + (17-2) + 3/5)] / 4 = 26/4 = 6.5 ملي ثانية

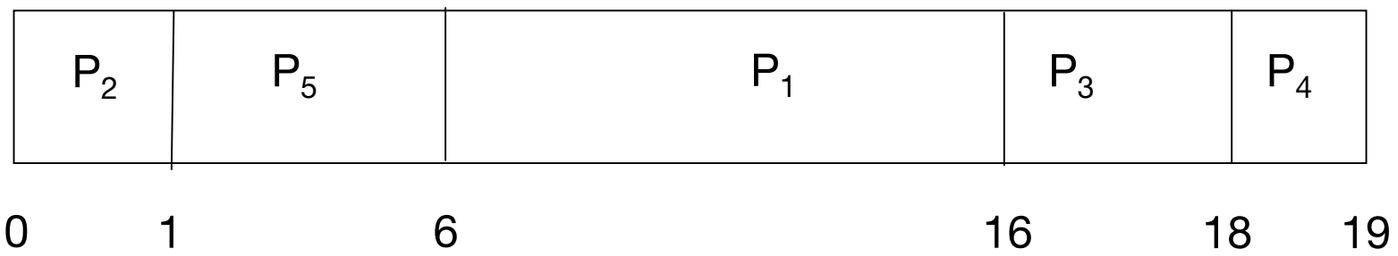
جدولة الأولويات

- مع كل عملية (صحيح)ويرتبط الاولوية رقم
- (الأولوية القصوى ≡أصغر عدد صحيح)يتم تخصيص وحدة المعالجة المركزية لعملية ذات الأولوية العليا
  - وقائي
  - غير وقائي
- هو جدولة الأولويات حيث الأولوية هي عكس المتوقع وحدة المعالجة المركزية في المرة القادمة انفجار SJF
- العمليات ذات الأولوية المنخفضة قد لا تنفذ -**مجاعة** ≡مشكلة
- مع مرور الوقت تقدم زيادة أولوية العملية -**شيخوخة** ≡حل

مثال على جدولة الأولوية

	وقت الانفجار	عملية	أفضلية
P_1	10	3	
P_2	1	1	
P_3	2	4	
P_4	1	5	
P_5	5	2	

- الأولوية جدولة مخطط جانت



- متوسط وقت الانتظار = 8.2 مللي ثانية

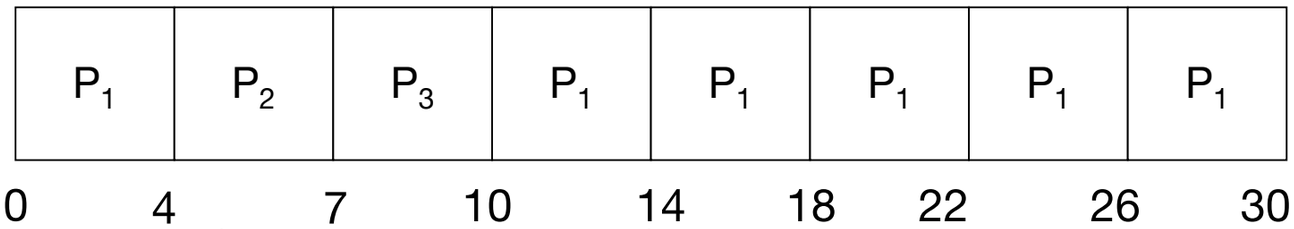
جولة روبن(RR)

- بعد انقضاء هذا الوقت، استبقت عملية .ميلي ثانية 100-10، وعادة (مرة كمية ف)كل عملية تحصل وحدة صغيرة من وقت وحدة المعالجة المركزية وتضاف إلى نهاية قائمة الانتظار جاهز.
- ن من وقت وحدة المعالجة المركزية في قطع من على / 1اذا كان هناك ن العمليات في قائمة انتظار استعداد والكم الوقت ف، ثم يحصل كل عملية ف وحدات زمنية(1-ن)أي عملية ينتظر أكثر من .الأكثر فوحدات الوقت في آن واحد.
- توقيت يقطع كل الكم لجدولة عملية المقبل
- أداء
  - فFIFO ⇒ كبير
  - صغير ف يجب أن تكون كبيرة فيما يتعلق تبديل السياق، فوق ذلك مرتفعة جدا ⇐ف

مثال RR مع الوقت الكم = 4

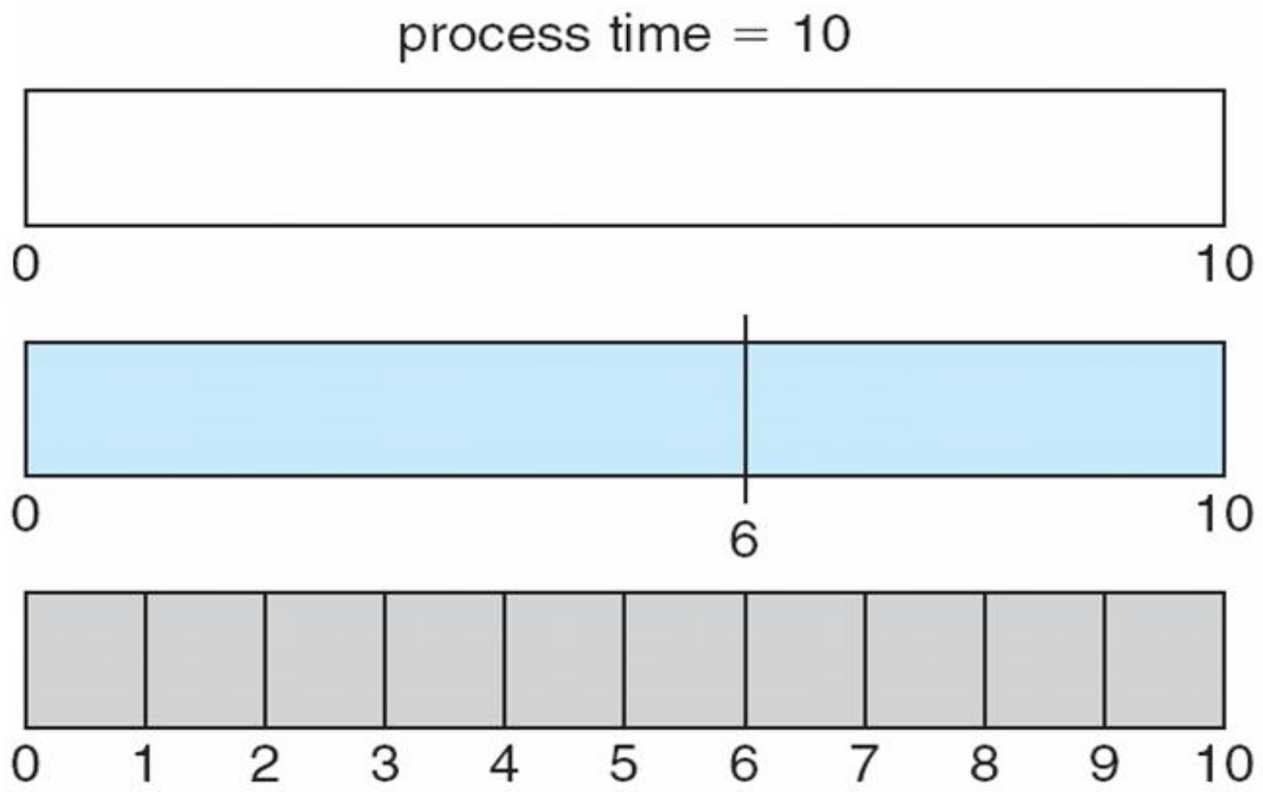
عملية	وقت الانفجار
P_1	24
P_2	3
P_3	3

- مخطط جانت هو:

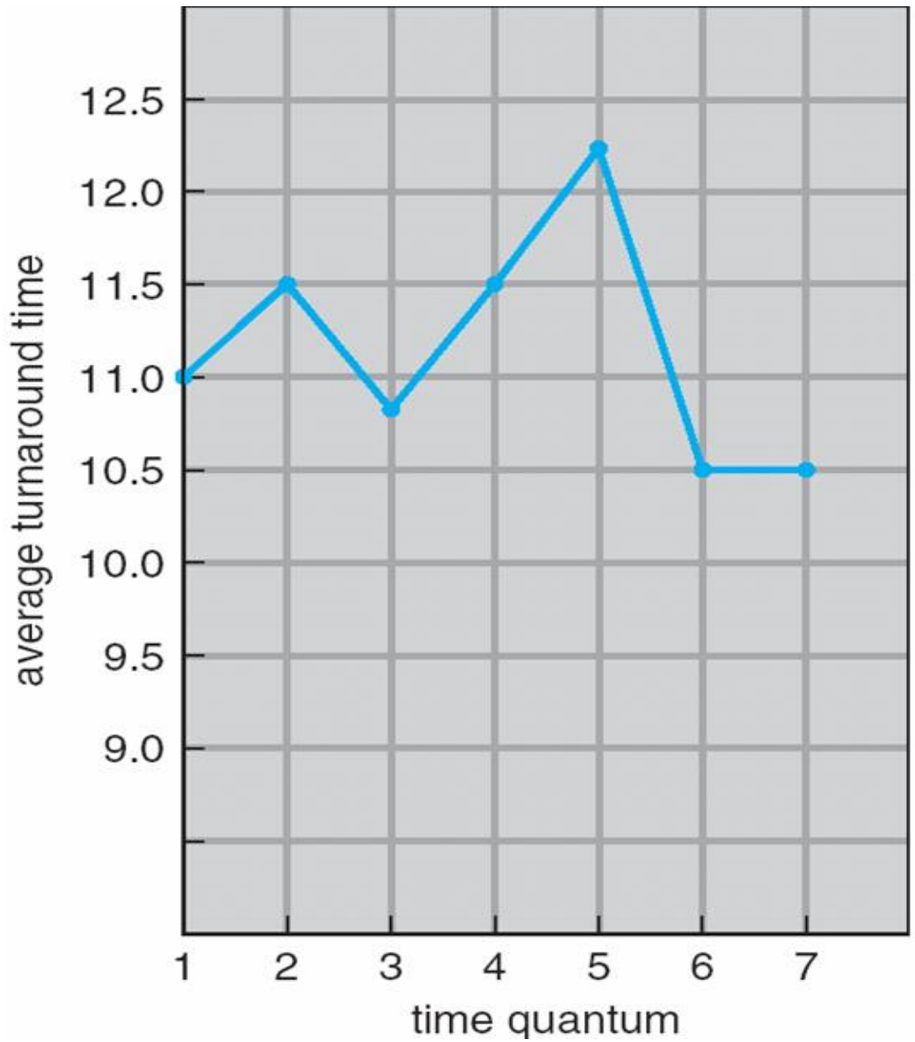


- ، ولكن أفضل *استجابة* SJFعادة، وهو أعلى معدل التحول من
- وينبغي أن يكون ف كبيرا بالمقارنة مع السياق وقت التبديل
- يوسيك 10>، تبديل سياق 100MS ل 10msف عادة

الوقت الكم والسياق تبديل الوقت



تحول الوقت يختلف مع
في الوقت الكم



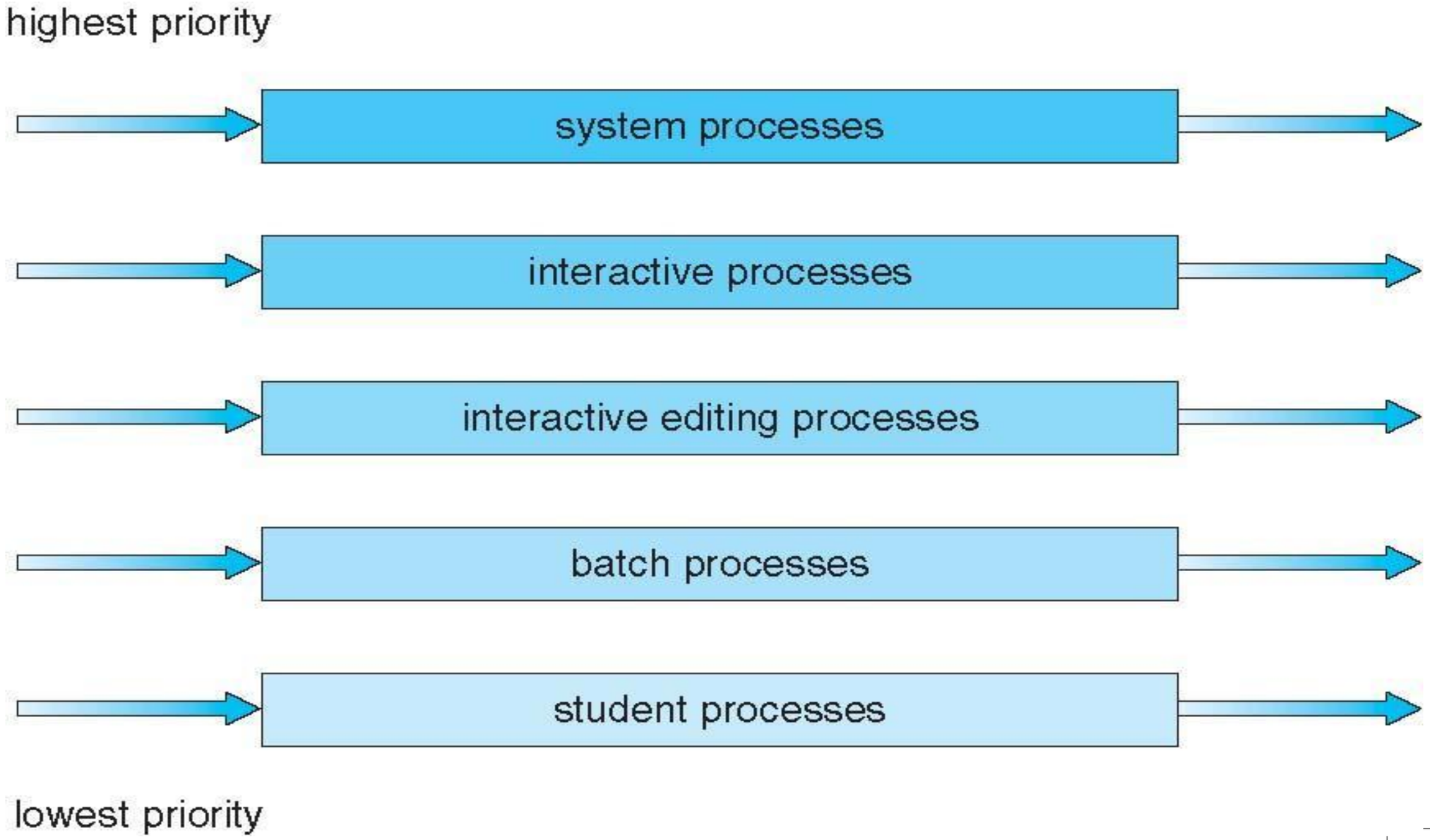
process	time
P_1	6
P_2	3
P_3	1
P_4	7

٪ من رشقات 80وينبغي أن يكون
نارية وحدة المعالجة المركزية أقصر من
ف

متعدد المستويات قائمة الانتظار

- يتم تقسيم الصف جاهزة في طوابير منفصلة، على سبيل المثال:
  - **المقدمة**(تفاعلية)
  - **خلفية**(دفعة)
- عملية دائمة في طابور معين
- كل طابور ديه جدولة الخوارزمية الخاصة بها:
  - المقدمة- RR
  - الخلفية- FCFS
- ويجب أن يتم جدولة بين قوائم الانتظار:
  - احتمال الموت جوعا .(أي خدمة جميع من المقدمة ثم من الخلفية)جدولة ثابتة ذات الأولوية؛
  - كل طابور يحصل على كمية معينة من الوقت وحدة المعالجة المركزية التي يمكن أن الجدول الزمني بين عملياتها -شريحة الوقت .أي 80 ٪ إلى الصدارة في RR
  - 20 ٪ لديهم خلفية في FCFS

متعدد المستويات قائمة الانتظار جدولة

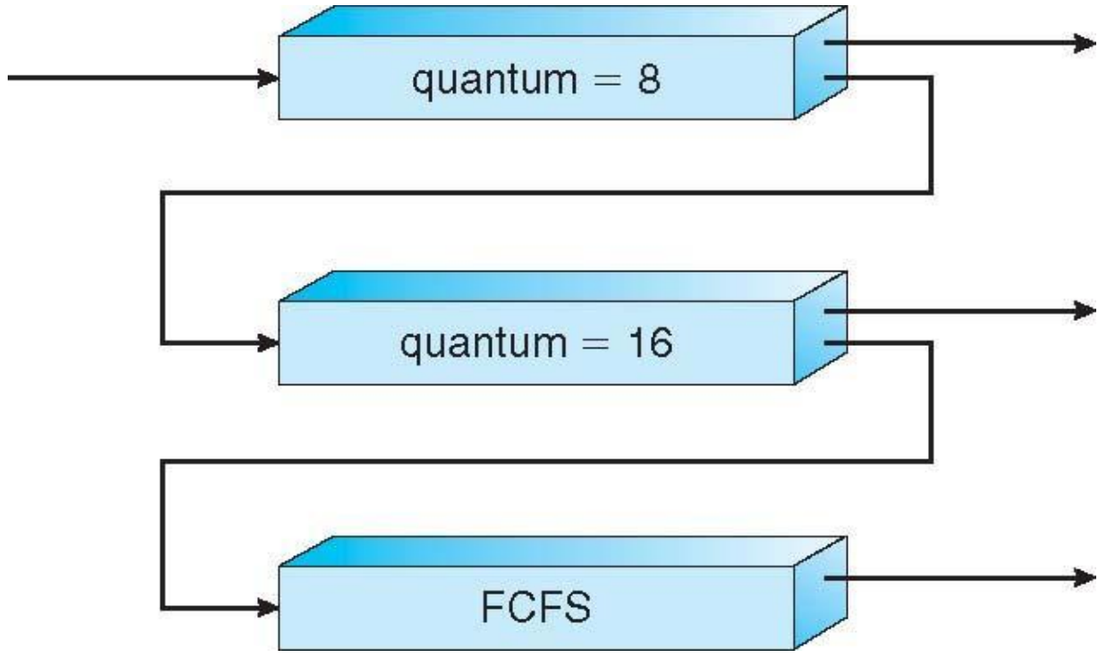


متعددة المستويات ملاحظات قائمة الانتظار

- يمكن تنفيذها بهذه الطريقة الشيخوخة .عملية يمكن ان تتحرك بين مختلف الطوابير
- جدولة متعددة المستويات ردود الفعل قائمة انتظار محددة من قبل المعلمات التالية:
  - عدد الصفوف
  - خوارزميات جدولة لكل قائمة انتظار
  - الطريقة المستخدمة لتحديد متى يتم ترقية عملية
  - الطريقة المستخدمة لتحديد متى يتم تخفيض عملية
  - الطريقة المستخدمة لتحديد قائمة الانتظار التي سوف عملية تدخل عندما تحتاج أن عملية الخدمة

مثال المتعدد المستويات ملاحظات قائمة الانتظار

- ثلاثة طوابير:
  - $س_0$ - RR الكم الوقت مع 8 ثانية ميلي
  - $س_1$ - RR الكم الوقت 16 ثانية ميلي
  - $س_2$ - FCFS
- جدولة
  - $س_0$ يدخل ألف وظيفة جديدة طابور FCFS الذي خدم
    - عندما يكسب وحدة المعالجة المركزية، وتتلقى وظيفة 8 ميلي ثانية
    - إذا كان لا ينتهي في 8 ميلي ثانية، يتم نقل الوظائف إلى قائمة الانتظار $س_1$
  - في $س_1$ يتم تقديم وظيفة ثانية 16 ميلي ثانية ويتلقى FCFS إضافية
    - إذا كان لا يزال لم يكمل، واستبقت ذلك وانتقل إلى قائمة الانتظار $س_2$



جدولة الموضوع

- التمييز بين مستوى المستخدم والمواضيع على مستوى النواة
- عندما المواضيع المعتمدة، المواضيع المقرر، وليس العمليات
- LWPكثير إلى واحد والعديد إلى العديد من النماذج، المواضيع جداول مكتبة موضوع على مستوى المستخدم لتشغيل على
  - منذ جدولة المنافسة في إطار عملية **(PCS)**المعروفة باسم **نطاق عملية خلاف**
  - يتم عادة عن طريق الأولوية التي وضعتها مبرمج
- المنافسة بين جميع المواضيع في نظام - **(SCS)**موضوع نواة المقرر على وحدة المعالجة المركزية المتاحة هو **نطاق نظام خلاف**

جدولة Pthread

- API يسمح بتحديد إما PCS أو SCS أثناء إنشاء موضوع
  - PTHREAD_SCOPE_PROCESS جدولة PCS جداول المواضيع باستخدام
  - PTHREAD_SCOPE_SYSTEM جدولة SCS جداول المواضيع باستخدام
- PTHREAD_SCOPE_SYSTEM السماح فقط OS X لينكس وماك -يمكن أن تقتصر من قبل نظام التشغيل

Pthread جدولةAPI

```
#  تشمل<pthread.h>
#  تشمل<stdio.h>
 تعريف# NUM  المواضيع5
 كثافة العمليات الرئيسية( الباحثARGC شار ،* ARGV []) {
كثافة العمليات ط، النطاق؛
   pthread  ر الدار[THREADS NUM].
   pthread ATTR  رATTR.
   / *الحصول على السمات الافتراضية  * /
   pthread ATTR  الحرف الأول(وATTR) ؛
   / *استفسار أولا على النطاق الحالي  * /
 إذا   (pthread ATTR getscope (& ATTRونطاق ،)! = 0)
     fprintf ( ستدير "غير قادر على الحصول على نطاق جدولة\ن") ؛
 آخر   {
 إذا       ( نطاق== PTHREAD سكوب)عملية
         printf ( " نطاق عمليةPTHREAD") ؛
 الا اذا      ( نطاق== PTHREAD سكوب)نظام
         printf ( "PTHREAD النظام نطاق") ؛
آخر
         fprintf ( ستدير "قيمة غير القانونية نطاق\ن.") ؛
   }
```

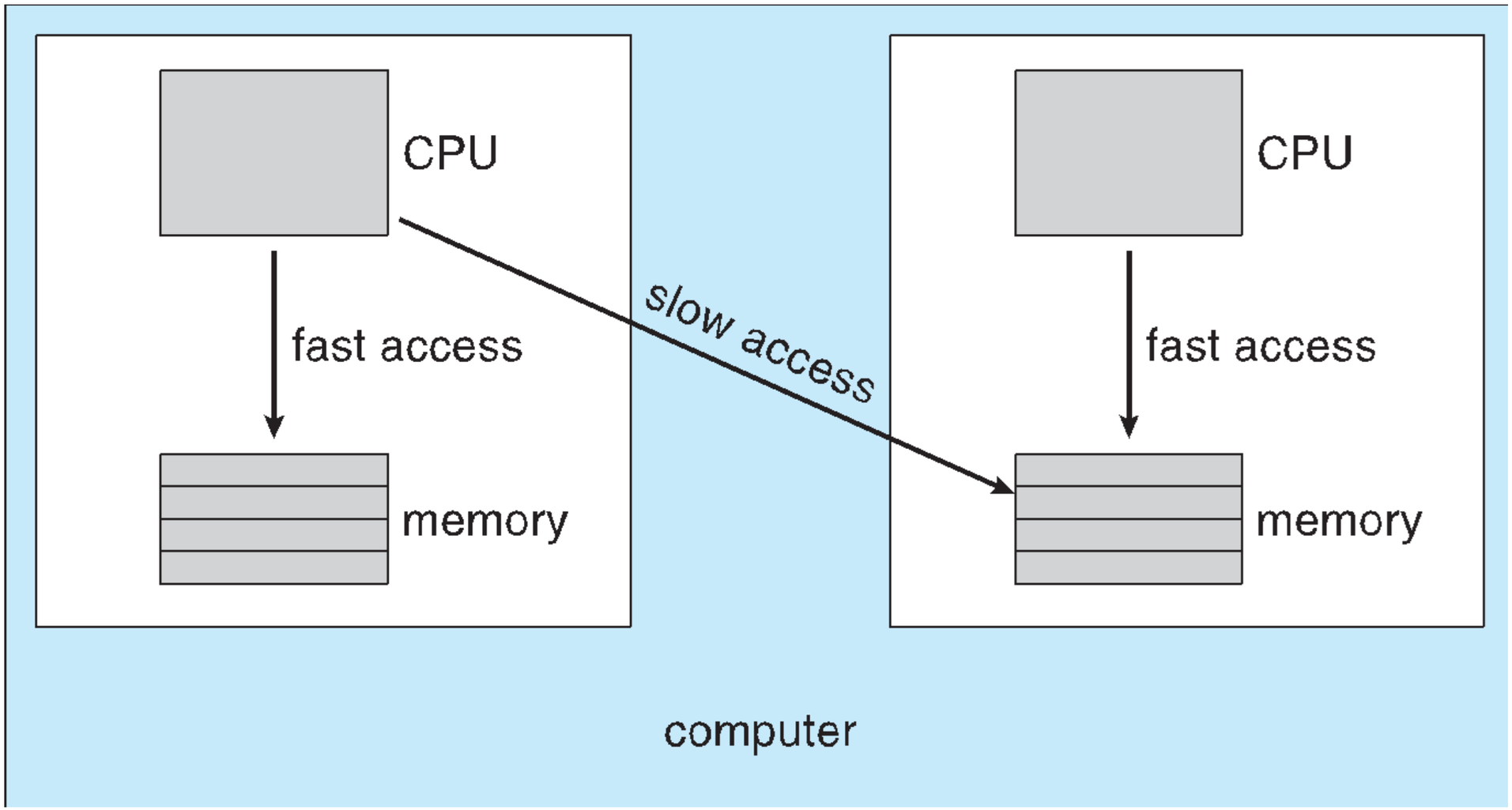
Pthread جدولةAPI

```
/ * SCSأو  PCSتعيين خوارزمية جدولة ل * /
pthread ATTR setscope (& ATTR ،PTHREAD نظام سكوب) ؛
/ *خلق المواضيع  * /
ل   ( ط= 0 ط ؛<NUM  وأنا الخيوط، ++)
    pthread  خلق( والدار[أنا]، ATTR، عداء، NULL) ؛
/ *الانضمام الآن على كل موضوع  * /
ل   ( ط= 0 ط ؛<NUM  وأنا الخيوط، ++)
    pthread  انضمام( الدار[أنا]، NULL) ؛
}
/ *وسيكون لكل موضوع يبدأ تحكم في هذه الوظيفة  * /
 *باطلة عداء( *الفراغ المعلمة)
{
  /*  القيام ببعض الأعمال... */
 خروج  pthread (0) ؛
}
```

جدولة متعددة المعالج

- جدولة وحدة المعالجة المركزية أكثر تعقيدا عندما تكون وحدات المعالجة المركزية متعددة متاحة
- **متجانس معالجات** ضمن المعالجات المتعددة
- **متعدد المعالجة غير المتماثلة** - معالج واحد فقط بالوصول إلى هياكل بيانات النظام، والتخفيف من الحاجة إلى تبادل البيانات
- **التوازي (SMP)** - كل المعالج هو جدولة النفس، جميع العمليات في طابور استعداد المشترك، أو كل لديه قائمة الانتظار خاص من العمليات جاهزة
  - حاليا، الأكثر شيوعا
- **بترابط المعالج** - عملية ديها تقارب للمعالج الذي كان يعمل في الوقت الحالي
  - **تقارب لينة**
  - **تقارب الصعب**
  - بما في ذلك الاختلافات **مجموعات معالج**

نوما وحدة المعالجة المركزية جدولة



لاحظ أن الخوارزميات الذاكرة وضع يمكن أن تنظر أيضا في تقارب

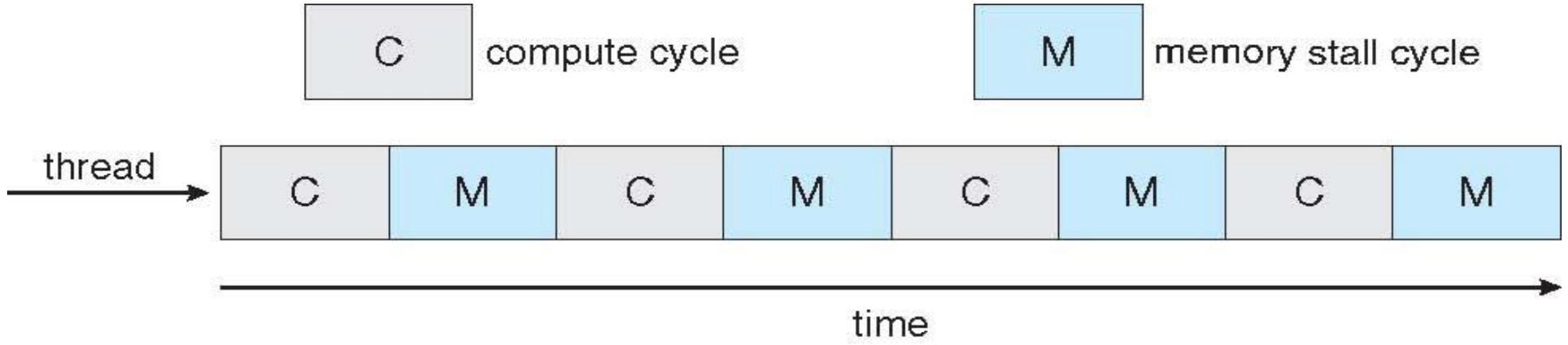
تحميل موازنة -جدولة متعددة المعالج

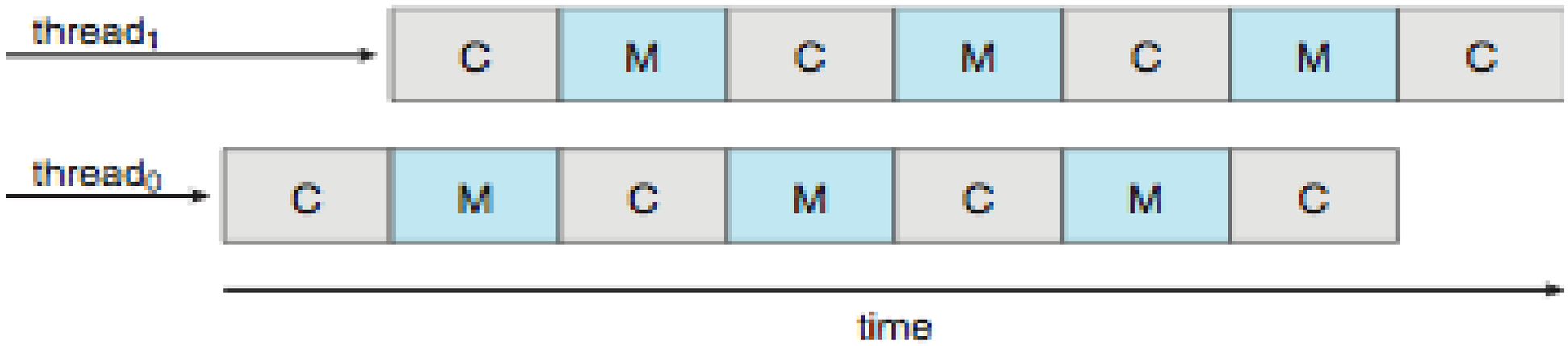
- إذا SMP، بحاجة للحفاظ على جميع وحدات المعالجة المركزية تحميلها للكفاءة
- **تحميل موازنة** محاولات للحفاظ على حجم العمل موزعة بالتساوي
- تحميل الشيكات مهمة دورية في كل معالج، وإذا وجدت مهمة الدفعات من وحدة المعالجة المركزية مثقلة إلى وحدات المعالجة المركزية الأخرى -**الهجرة دفع**
- المعالجات الخمول تسحب ينتظر مهمة من معالج مشغول -**الهجرة سحب**

والمتعددة المعالجات

- الاتجاه الأخير لوضع النوى المتعددة في المعالجات على رقاقة المادي نفسه
- أسرع وتستهلك طاقة أقل
- مواضيع متعددة لكل نواة ينمو أيضا
  - يستفيد من المماطلة الذاكرة لإحراز تقدم في موضوع آخر في حين يحدث الذاكرة استرداد

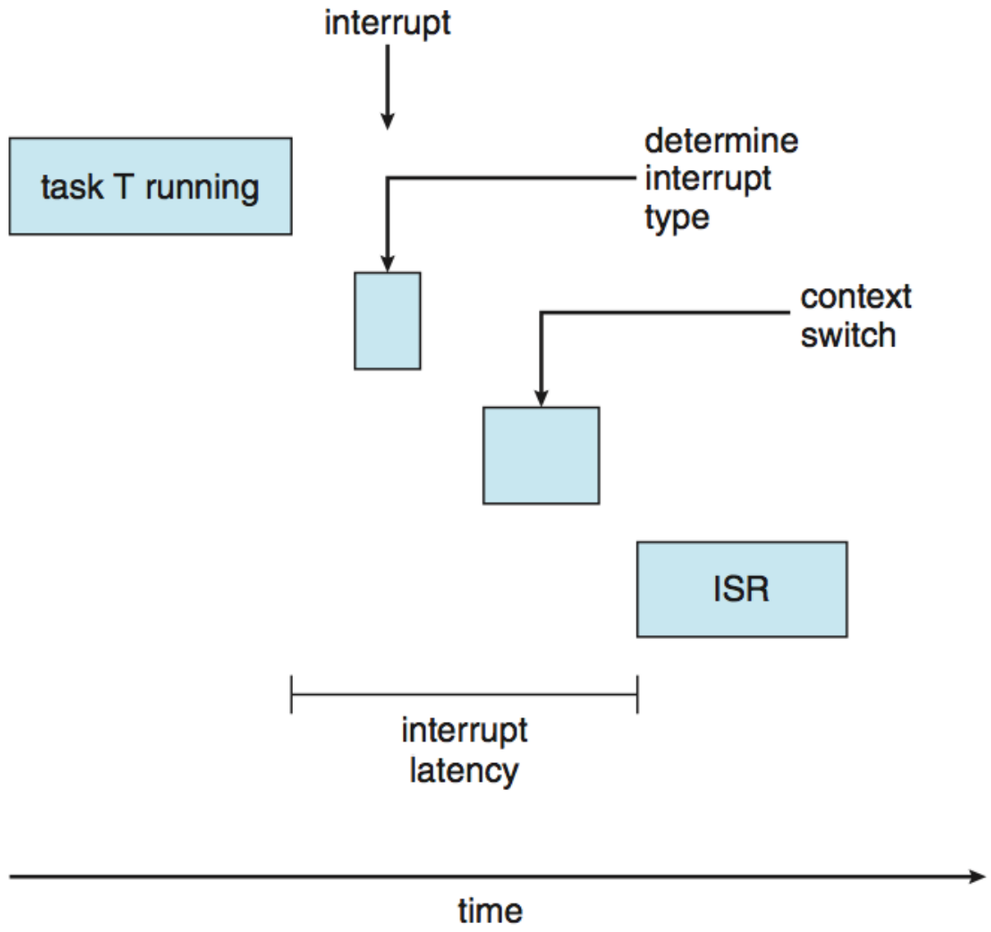
نظام متعددة النواة مؤشرات





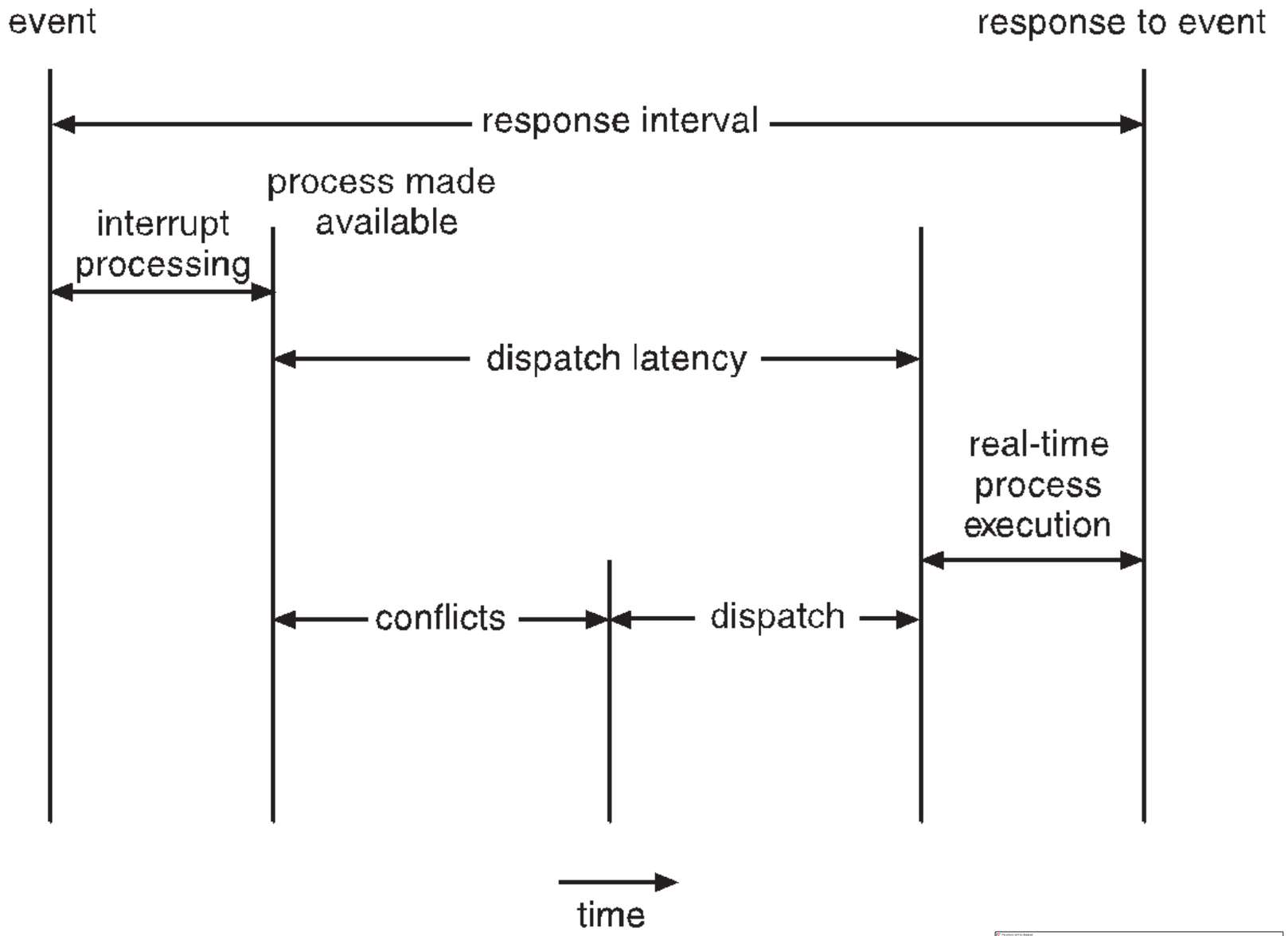
في الوقت الحقيقي وحدة المعالجة المركزية جدولة

- يمكن أن تمثل تحديات واضحة
- **أنظمة الوقت الحقيقي لينة** - لن تعقد ضمان لعملية في الوقت الحقيقي عندما حرجة
- **أنظمة الوقت الحقيقي الصعبة** - جب أن تكون الخدمات المهمة بحلول الموعد النهائي المتمثل
- نوعين من الإختفاء تؤثر على الأداء
  1. زمن المقاطعة - الساعة من وصول المقاطعة لبدء من الروتين الذي يقطع الخدمات
  2. الوقت لجدول زمني لاتخاذ العملية - إرسال الكمون الحالية من وحدة المعالجة المركزية والتحول إلى أخرى



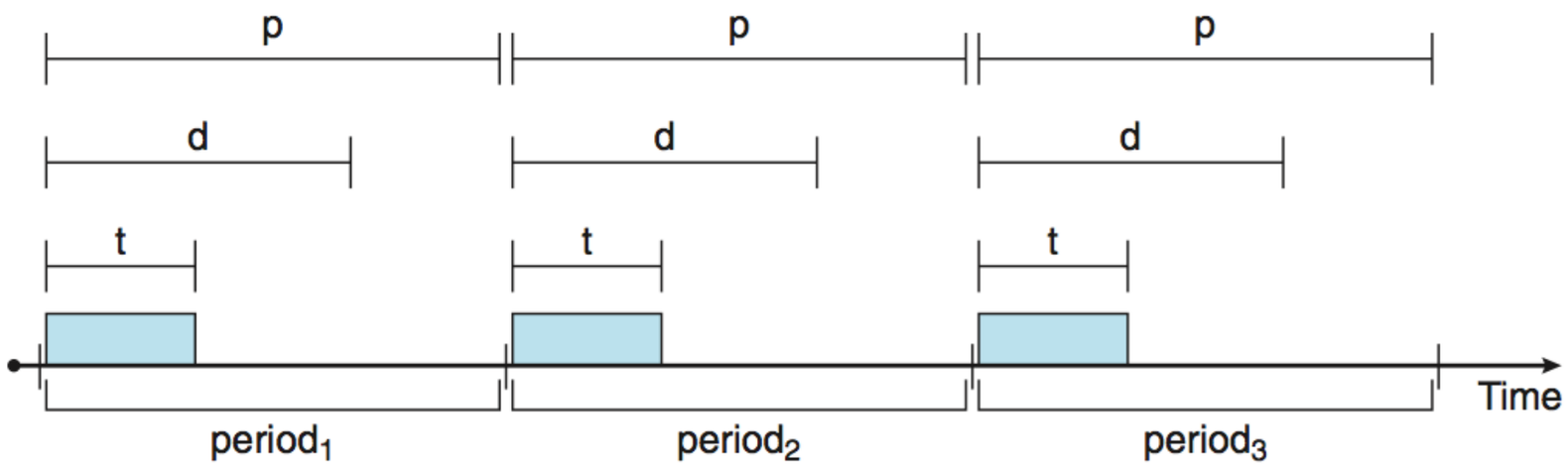
(يتبع)في الوقت الحقيقي وحدة المعالجة المركزية جدولة

- مرحلة الصراع من الكمون ارسال:
  1. الشفعة في أي عملية قيد التشغيل في وضع النواة
  2. الإفراج عن طريق عملية ذات الأولوية المنخفضة الموارد التي تحتاجها العمليات ذات الأولوية العالية



جدولة القائم على أولوية

- لجدولة الوقت الحقيقي، يجب جدولة دعم وقائية، وجدولة القائم على أولوية
  - ولكن فقط يضمن لينة في الوقت الحقيقي
- يجب أن لمن الصعب في الوقت الحقيقي أيضا توفير القدرة على الوفاء بالمواعيد النهائية
- **دوري** تتطلب منها وحدة المعالجة المركزية على فترات ثابتة :العمليات لها خصائص جديدة
  - لديها وقت المعالجة *ت*، الموعد النهائي *د*، فترة *ص*
  - $0 \leq ت \leq د \leq ص$
  - **معدل** مهمة دورية هو 1/ *ص*

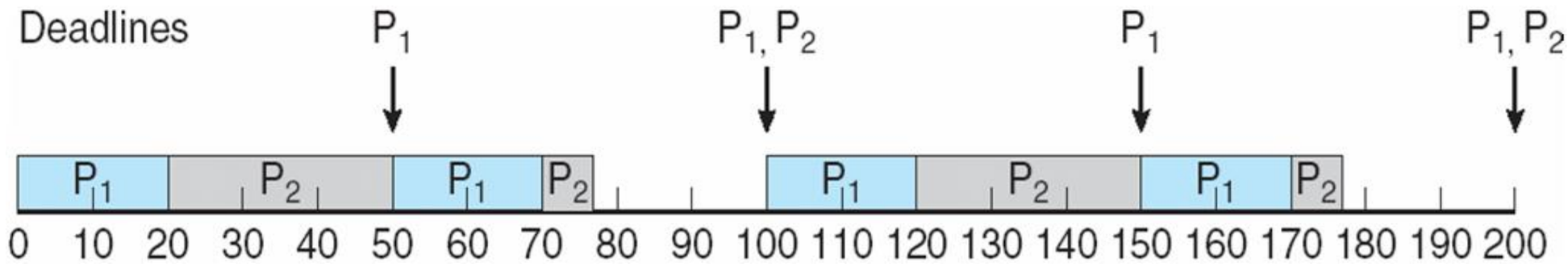


الافتراضية وجدولة

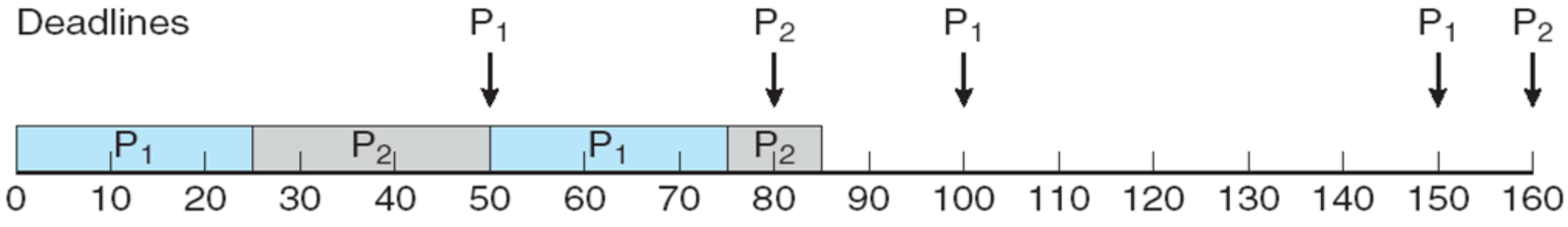
- (الصورة)جداول البرمجيات الافتراضية الضيوف متعددة على وحدة المعالجة المركزية
- كل ضيف القيام الجدولة الخاصة
  - لا يعرفون أنها لا تملك وحدات المعالجة المركزية
  - يمكن أن يؤدي ذلك إلى ضعف زمن الاستجابة
  - يمكن أن تؤثر على الوقت من اليوم في الساعات الضيوف
- يمكنك التراجع جيدة جهود جدولة الخوارزمية الضيوف

جدولة Montonicمعدل

- يتم تعيين يستند الأولوية على عكس فترتها
- أولوية أعلى =فترات أقصر.
- أولوية أقل =فترات أطول
- P_1 يتم تعيين أولوية أعلى من P_2.



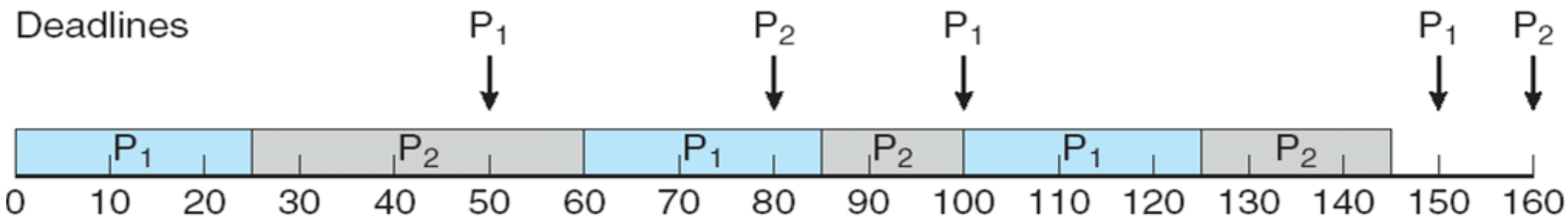
عدم الوفاء بالمواعيد المحددة مع معدل المطردة جدولة



أقرب الموعد النهائي الأولى جدولة(EDF)

- يتم تعيين الأولويات وفقا لمواعيد نهائية:

  في وقت سابق من الموعد المحدد، وارتفاع الأولوية؛
  في وقت لاحق الموعد المحدد، وانخفاض الأولوية



تتناسب حصة جدولة

- ت يتم توزيع الحصص بين كافة العمليات في نظام
- يتلقى تطبيق *N* سهم حيث *N* < *T*
- ت من إجمالي وقت المعالج / ***N*** وهذا يضمن كل تطبيق سيحصل

في الوقت الحقيقي POSIXجدولة

- المعيارPOSIX.1b
- وظائف لإدارة المواضيع في الوقت الحقيقي APIيوفر
- تعرف فئتين جدولة للمواضيع في الوقت الحقيقي:

1. ليس هناك وقت تشريح للمواضيع ذات أولوية .SCHED_FIFO - مع قائمة انتظار يخرج أولا FCFSومن المقرر المواضيع باستخدام استراتيجية متساوية
2. SCHED_RR - إلا مرة وتشريح يحدث للمواضيع ذات أولوية متساوية SCHED_FIFOعلى غرار

- تعرف ظيفتين من أجل الحصول على ووضع سياسة الجدولة:

1. **pthread ATTR getsched سياسة(pthread ATTR ر* ATTR السياسات الباحث ،*)**
2. **pthread ATTR سياسةsetsched (pthread ATTR ر* ATTRسياسة دولية ،)**

جدولة APIفي الوقت الحقيقي POSIX

```
#  تشمل<pthread.h>
#  تشمل<stdio.h>
 تعريف# NUM  المواضيع5
 الباحث الرئيسي( الباحثARGC شار ،* ARGV [])
{
كثافة العمليات ط، السياسات؛
   pthread  ر الدار[THREADS NUM].
   pthread ATTR  رATTR.
   / *الحصول على السمات الافتراضية  * /
  pthread ATTR  الحرف الأول(وATTR) ؛
  / *الحصول على سياسة الجدولة الحالية  * /
إذا   (pthread ATTR getschedpolicy (& ATTRوالسياسة ،)! = 0)
    fprintf ( ستدير "غير قادر على الحصول سياسة \ن.") ؛
آخر   {
إذا       ( السياسة==قائع أخرى) printf ( " قائع أخرى\ن") ؛
الا اذا      ( السياسة== SCHED RR) printf ( "RR SCHED \ ن") ؛
الا اذا      ( السياسة== FIFO SCHED) printf ( "FIFO SCHED \ ن") ؛
  }
```

(يتبع) API في الوقت الحقيقي جدولة POSIX

```
   / *  تعيين سياسة الجدولة- FIFO ، أو RR ، أخرى  * /
 إذا   (pthread ATTR setschedpolicy (& ATTR ، FIFO SCHED)! = 0)
      fprintf ( ستدير "غير قادر على تعيين سياسة \ن.") ؛
   / *خلق المواضيع  * /
ل   ( ط= 0 ط ؛<NUM  وأنا الخيوط، ++)
      pthread  خلق( و[أنا]والدار ، ATTR ، عداء ، NULL) ؛
   / *الانضمام الآن على كل موضوع  * /
ل   ( ط= 0 ط ؛<NUM  وأنا الخيوط، ++)
      pthread  انضمام( [أنا]الدار ، NULL) ؛
}

/ *وسيكون لكل موضوع يبدأ تحكم في هذه الوظيفة  * /
 (*الفراغ المعلمة )عداء  *باطلة
{
   /*  القيام ببعض الأعمال... */
 خروج   pthread (0) ؛
}
```

أمثلة نظام التشغيل

- لينكس جدولة
- نوافذ جدولة
- سولاريس جدولة

2.5لينكس جدولة من خلال الإصدار

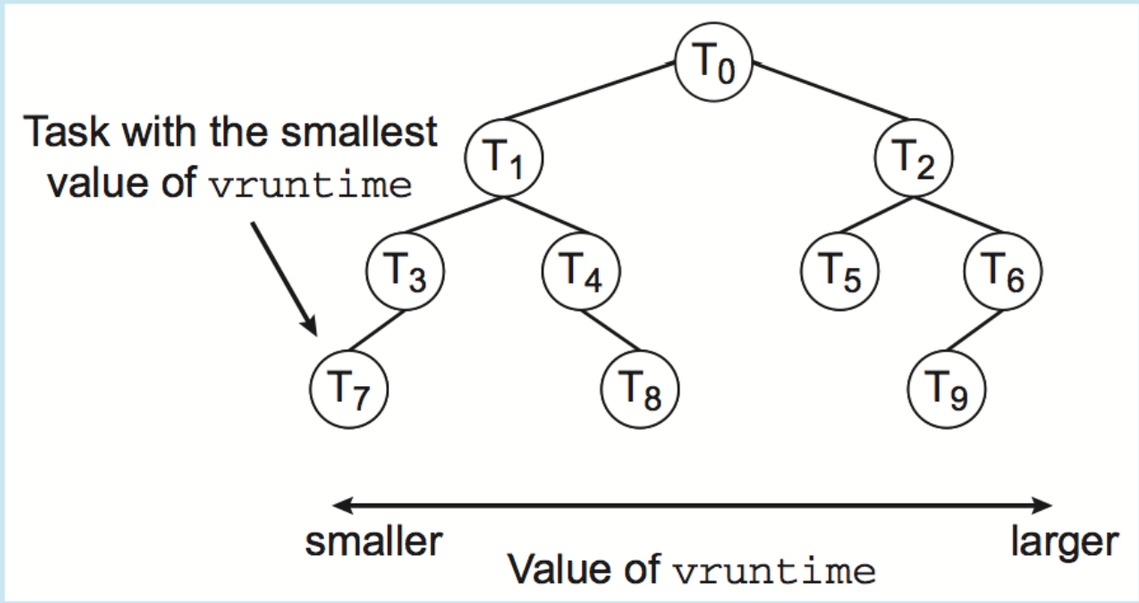
- n ، ركض تباين القياسية يونيكس جدولة الخوارزمية2.5قبل نواة الإصدار
- n جدولة الوقت (1)Oانتقل الى ترتيب ثابت 2.5الإصدار
  - l وقائي، أولوية قائم
  - l تقاسم الوقت والوقت الحقيقي :وتتراوح مدة الأولوية
  - l 140-100و **لطيف** قيمة 99-0**في الوقت الحالى** تتراوح
  - l تعيين في أولوية عالمية مع انخفاض القيم العددية تشير إلى أولوية أعلى
  - l أولوية أعلى يحصل ف أكبر
  - l (**نشط**)تشغيل، وقادر على العمل طالما الوقت المتبقي في شريحة الوقت
  - l ، وليس التشغيل قادرة حتى تستخدم كل المهام الأخرى شرائح بهم(**منتهية الصلاحية**)إذا لم يكن هناك الوقت المتبقي
  - l هيكل البيانات **runqueue**جميع المهام المدى قادرة تعقبها في كل وحدة المعالجة المركزية
    - 4 (نشط، انتهت)اثنين من صفائف الأولوية
    - 4 المهام فهرستها حسب الأولوية
    - 4 عندما لا أكثر نشاطا، ويتم تبادل صفائف
  - l عملت بشكل جيد، ولكن مرة ضعف الاستجابة لعمليات التفاعلية

2.6.23 + جدولة لينكس في النسخة

- ***جدولة تماما العادلة***(CFS)
- **دروس جدولة**
  - كل له أولوية محددة
  - جدولة يختار أعلى المهمة ذات الأولوية في أعلى فئة جدولة
  - بدلا من الكم على أساس المخصصات زمنية محددة، على أساس نسبة من وقت وحدة المعالجة المركزية
  - وشملت فئات جدولة، والبعض الآخر يمكن أن تضاف 2
    1. الافتراضي
    2. في الوقت الحالى
- 19+إلى 20-الكم محسوبة على أساس **قيمة لطيفة** من
  - أقل قيمة هي أولوية أعلى
  - فترة من الزمن خلالها يجب تشغيل المهمة مرة واحدة على الأقل -تحسب **الكمون الهدف**
  - الهدف الكمون يمكن أن تزيد إذا نقول ازدياد عدد المهام النشطة
- **وقت التشغيل الظاهري** ثابت **`vruntime`**وتؤكد لجنة الأمن الغذائي جدولة لكل مهمة
  - أولوية أدنى هو أعلى معدل تسوس -يرتبط مع عوامل الاضمحلال على أساس أولوية مهمة
  - وقت التشغيل الفعلي =الأولوية الافتراضي طبيعية ينتج وقت التشغيل الظاهري
- أن تقرر المهمة التالية لتشغيل، مهمة يختار جدولة بأقل وقت التشغيل الظاهري

الأداء جنة الأمن الغذائي العالمي

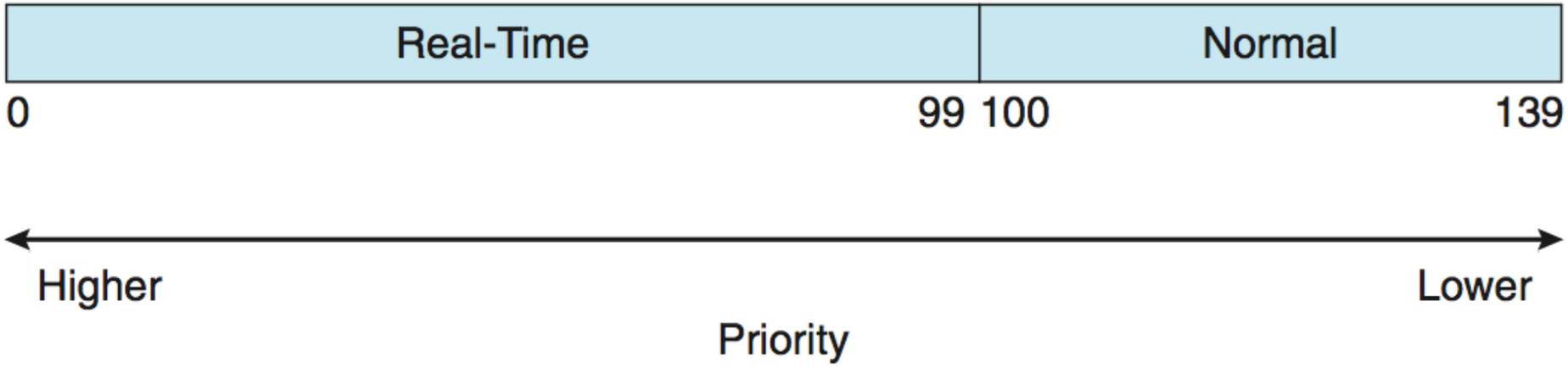
The Linux CFS scheduler provides an efficient algorithm for selecting which task to run next. Each runnable task is placed in a red-black tree—a balanced binary search tree whose key is based on the value of `vruntime`. This tree is shown below:



When a task becomes runnable, it is added to the tree. If a task on the tree is not runnable (for example, if it is blocked while waiting for I/O), it is removed. Generally speaking, tasks that have been given less processing time (smaller values of `vruntime`) are toward the left side of the tree, and tasks that have been given more processing time are on the right side. According to the properties of a binary search tree, the leftmost node has the smallest key value, which for the sake of the CFS scheduler means that it is the task with the highest priority. Because the red-black tree is balanced, navigating it to discover the leftmost node will require $O(lg N)$ operations (where N is the number of nodes in the tree). However, for efficiency reasons, the Linux scheduler caches this value in the variable `rb_leftmost`, and thus determining which task to run next requires only retrieving the cached value.

(يتبع)لينكس جدولة

- جدولة الوقت الحقيقي وفقا لPOSIX.1b
  - المهام في الوقت الحقيقي ديها أولويات ثابتة
- في الوقت الحقيقي بالإضافة إلى الخريطة العادية في مخطط الأولوية العالمي
- 100إلى أولوية عالمية 20-قيمة لطيفة من الخرائط
- 139خرائط لأولوية 19+قيمة لطيفة من



نوافذ جدولة

- القائم على أولوية تحديد مواعيد وقائية Windowsيستخدم نظام التشغيل
- موضوع أولوية أعلى يدير المقبل
- **المرسل** هو مجدول
- استبقت التي كتبها العالي ذات الأولوية موضوع (3)يستخدم شريحة الوقت، (2)كتل، (1)يعمل الموضوع حتى
- يمكن المواضيع في الوقت الحقيقي استباق الوقت غير الحقيقي
- 32مخطط الأولوية على مستوى
- 16-31، **الطبقة الوقت الحقيقي** هو 15/01**الطبقة متغير** هو
- هو موضوع إدارة الذاكرة 0الأولوية
- قائمة الانتظار لكل أولوية
- إذا لم يكن هناك موضوع المدى قادرة، تدير **موضوع الخمول**

فئات الأولوية ويندوز

- واجهة برمجة التطبيقات عدة فئات الأولوية التي يمكن للعملية تنتمي Win32ويحدد
  - REALTIME_PRIORITY_CLASS ،HIGH_PRIORITY_CLASS ،ABOVE_NORMAL_PRIORITY_CLASS ، NORMAL_PRIORITY_CLASS ،BELOW_NORMAL_PRIORITY_CLASS ،IDLE_PRIORITY_CLASS
  - REALTIMEكلها متغيرة إلا
- وموضوع ضمن فئة إعطاء الأولوية لديه الأولوية النسبية
  - TIME_CRITICAL ،أعلى ABOVE_NORMAL ،NORMAL ،BELOW_NORMAL ،أدنى IDLE
- فئة الأفضلية والأولوية النسبية تجتمع لتعطي الأولوية رقمية
- الأولوية الأساسية هي عادي داخل الطبقة
- في حال انتهاء صلاحية الكم، والأولوية خفضت، ولكن أبدا دون قاعدة
- إذا حدث الانتظار، عززت الأولوية اعتمادا على ما كان ينتظر ل
- التحسين الأولوية 3Xالإطار الأمامي نظرا
- (ومس)جدولة وضع المستخدم 7وأضاف ويندوز
  - تطبيقات إنشاء وإدارة المواضيع المستقلة النواة
  - لعدد كبير من المواضيع وأكثر كفاءة بكثير
  - الإطار (ConcRT)وقت التشغيل المتزامن C ++ومس المنظمون تأتي من المكتبات لغة البرمجة مثل

الأولويات ويندوز

	real-time	high	above normal	normal	below normal	idle priority
time-critical	31	15	15	15	15	15
highest	26	15	12	10	8	6
above normal	25	14	11	9	7	5
normal	24	13	10	8	6	4
below normal	23	12	9	7	5	3
lowest	22	11	8	6	4	2
idle	16	1	1	1	1	1

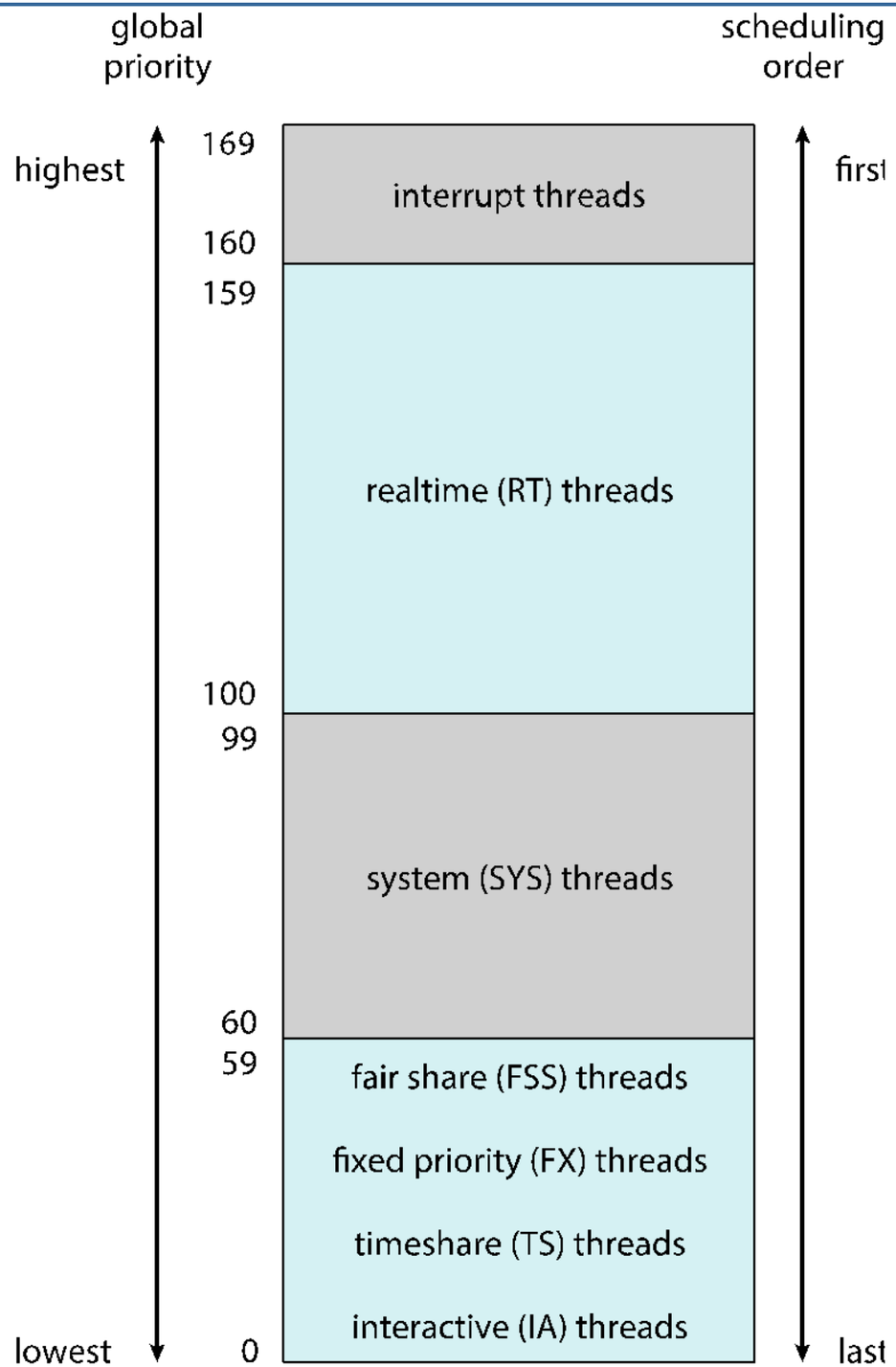
سولاريس

- جدولة القائم على أولوية
- ستة فصول المتاحة
  - اقتسام الوقت (الافتراضي)(TS)
  - التفاعلية(IA)
  - في الوقت الحقيقي(RT)
  - نظام(SYS)
  - عادلة حصة(FSS)
  - أولوية ثابتة(اف ب)
- موضوع معين يمكن أن يكون في فئة واحدة في وقت واحد
- كل فئة لديها جدولة الخوارزمية الخاصة
- اقتسام الوقت هو متعدد المستويات انتظار ردود الفعل
  - شكلي من قبل مسؤول النظام Loadableالجدول

سولاريس مرسل الجدول

priority	time quantum	time quantum expired	return from sleep
0	200	0	50
5	200	0	50
10	160	0	51
15	160	5	51
20	120	10	52
25	120	15	52
30	80	20	53
35	80	25	54
40	40	30	55
45	40	35	56
50	40	40	58
55	40	45	58
59	20	49	59

سولاريس جدولة



(يتبع)سولاريس جدولة

- جدولة تحويل أولويات فئة محددة إلى أولوية عالمية لكل موضوع
  - الموضوع مع أولوية قصوى يعمل المقبل
  - ومتأثرة أعلى أولوية موضوع (3)يستخدم شريحة الوقت، (2)كتل، (1)يستمر حتى
  - RRالمواضيع متعددة في نفس الأولوية المختارة عبر

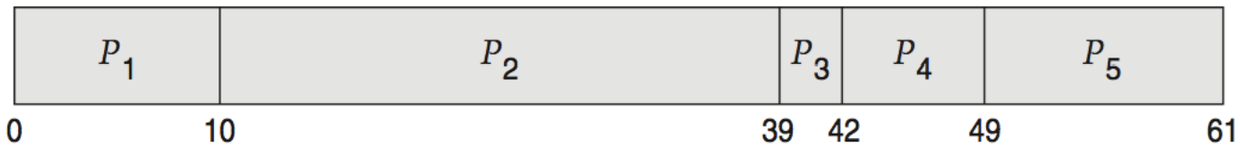
تقييم خوارزمية

- كيفية اختيار خوارزمية وحدة المعالجة المركزية جدولة لنظام التشغيل؟
- تحديد المعايير، ثم تقييم الخوارزميات
- **النمذجة القطعية**
  - نوع من **التقييم التحليلي**
  - يأخذ عبء العمل المحددة مسبقا معين ويحدد أداء كل خوارزمية لأن عبء العمل
- عمليات وصوله في وقت 5 تنظر 0:

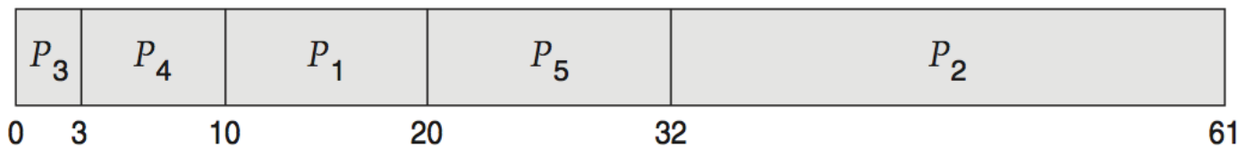
Process	Burst Time
P_1	10
P_2	29
P_3	3
P_4	7
P_5	12

تقييم حتمية

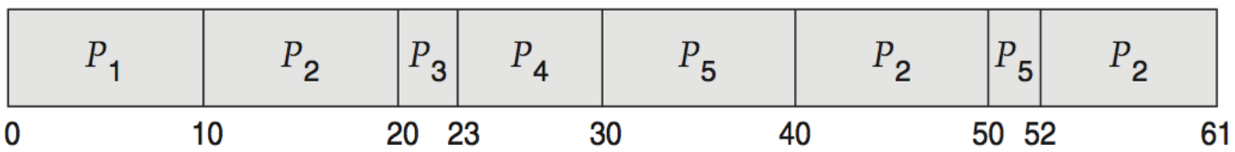
- لكل خوارزمية حساب الحد الأدنى متوسط وقت الانتظار
- بسيطة وسريعة، ولكنها تتطلب العدد الدقيق للمدخلات، لا ينطبق إلا على تلك المدخلات
  - FCS هو28ms:



  - غير استباقية SPJ هو 13ms و:



  - RR هو23MS:



تشكيل طابور نماذج

- رشقات نارية احتماليا I / O ايصف وصول العمليات، وحدة المعالجة المركزية و
  - الأسي شيوعا، والتي وصفها المتوسط
  - يحسب معدل الإنتاجية والاستخدام ووقت الانتظار، الخ
- وصف نظام الكمبيوتر عن شبكة من الخوادم، مع كل طابور من عمليات الانتظار
  - معرفة معدلات الوصول والأسعار الخدمة
  - يحسب الاستخدام، ومتوسط طول انتظار، ومتوسط وقت الانتظار، الخ

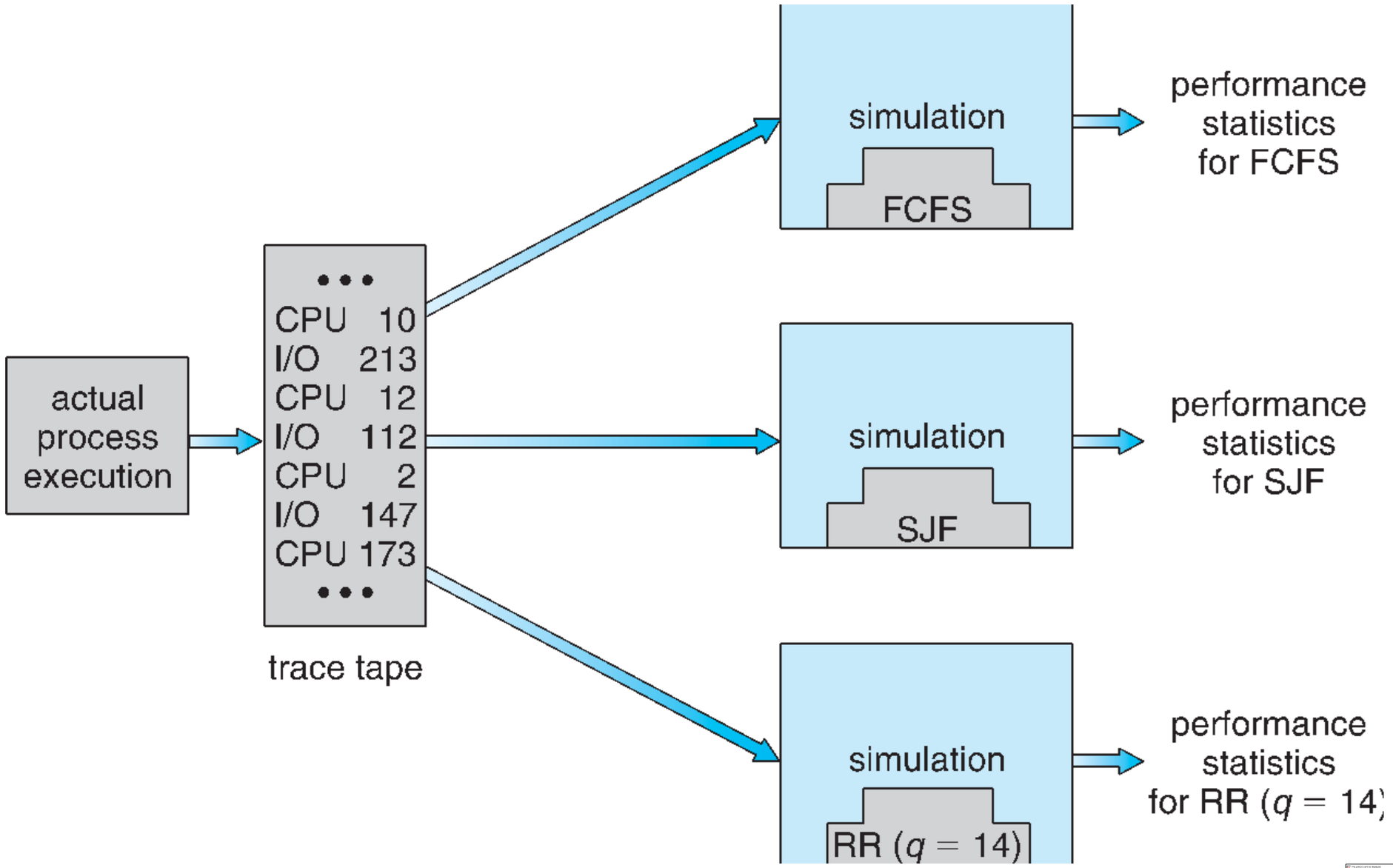
الفورمولا يتل

- متوسط طول انتظار =ن
- متوسط وقت الانتظار في الطابور = *W*
- متوسط معدل وصول في قائمة الانتظار = λ
- في حالة مستقرة، وعمليات مغادرة يجب انتظار عمليات متساوية وصوله، وبالتالي -القانون قليلا
  ن= λ س *W*
  - صالحة للأي خوارزمية جدولة وتوزيع وصوله
- ثانية 2 =عملية في قائمة الانتظار، ثم متوسط وقت الانتظار في عملية 14عمليات يصل في الثانية الواحدة، وعادة 7على سبيل المثال، إذا كان على متوسط

المحاكاة

- نماذج تشكيل طابور محدودة
- **المحاكاة** أكثر دقة
  - نموذج برمجة نظام الكمبيوتر
  - على مدار الساعة هو متغير
  - جمع الإحصاءات تشير إلى أداء الخوارزمية
  - البيانات لدفع محاكاة جمعها عبر
    - عدد مولد عشوائيا وفقا لاحتمالات
    - توزيعات محددة رياضيا أو تجريبيا
    - تتبع الأشرطة تسلسل سجل من أحداث حقيقية في أنظمة حقيقية

تقييم المنظمون وحدة المعالجة المركزية بواسطة المحاكاة



التنفيذ

- ودقة المحاكاة بل محدودة
- مجرد تنفيذ جدولة جديدة واختبار في النظم الحقيقية
  - ارتفاع تكلفة ومخاطر عالية
  - بيئات تختلف
- معظم المنظمون مرنة يمكن تعديلها لكل موقع أو لكل نظام
- أو واجهات برمجة التطبيقات لتعديل الأولويات
- ولكن مرة أخرى بيئات تختلف