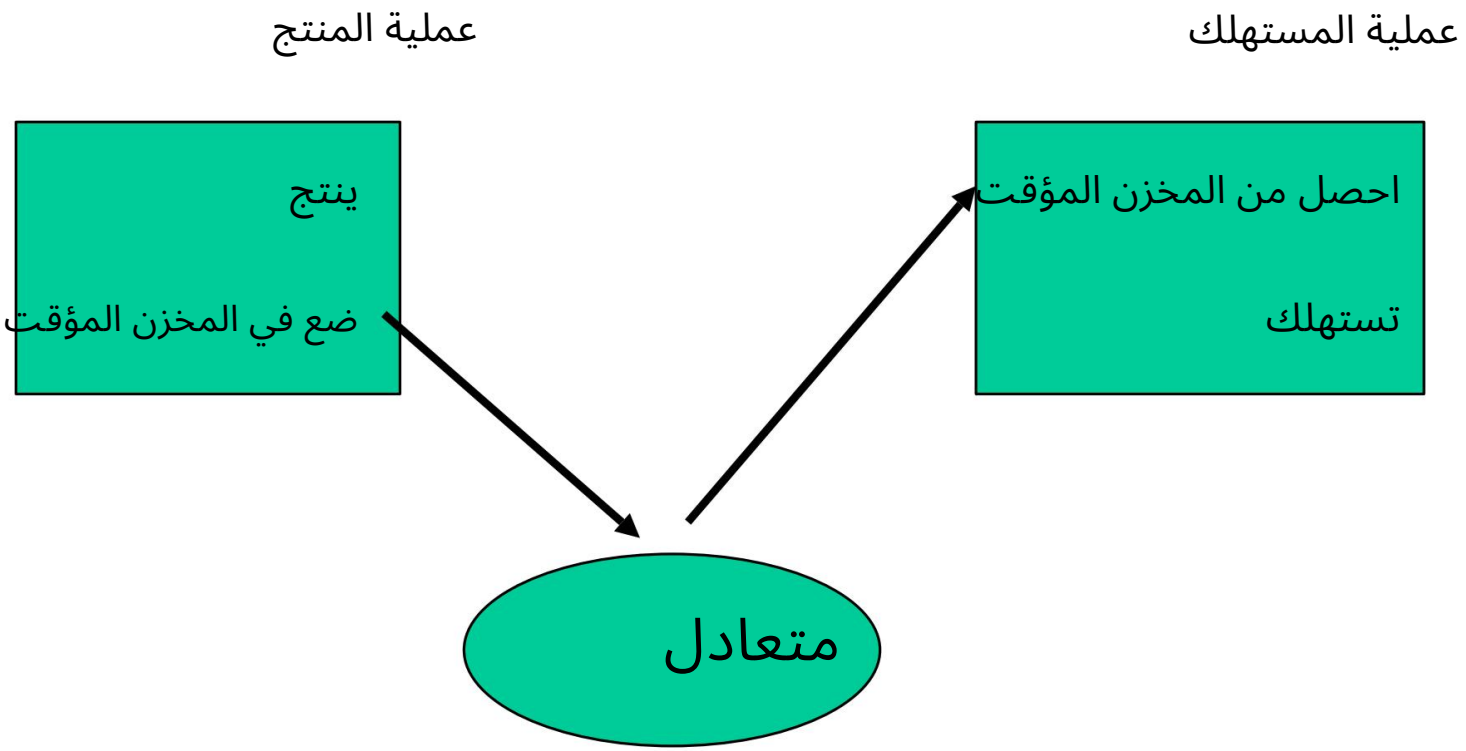
اتصال interprocess

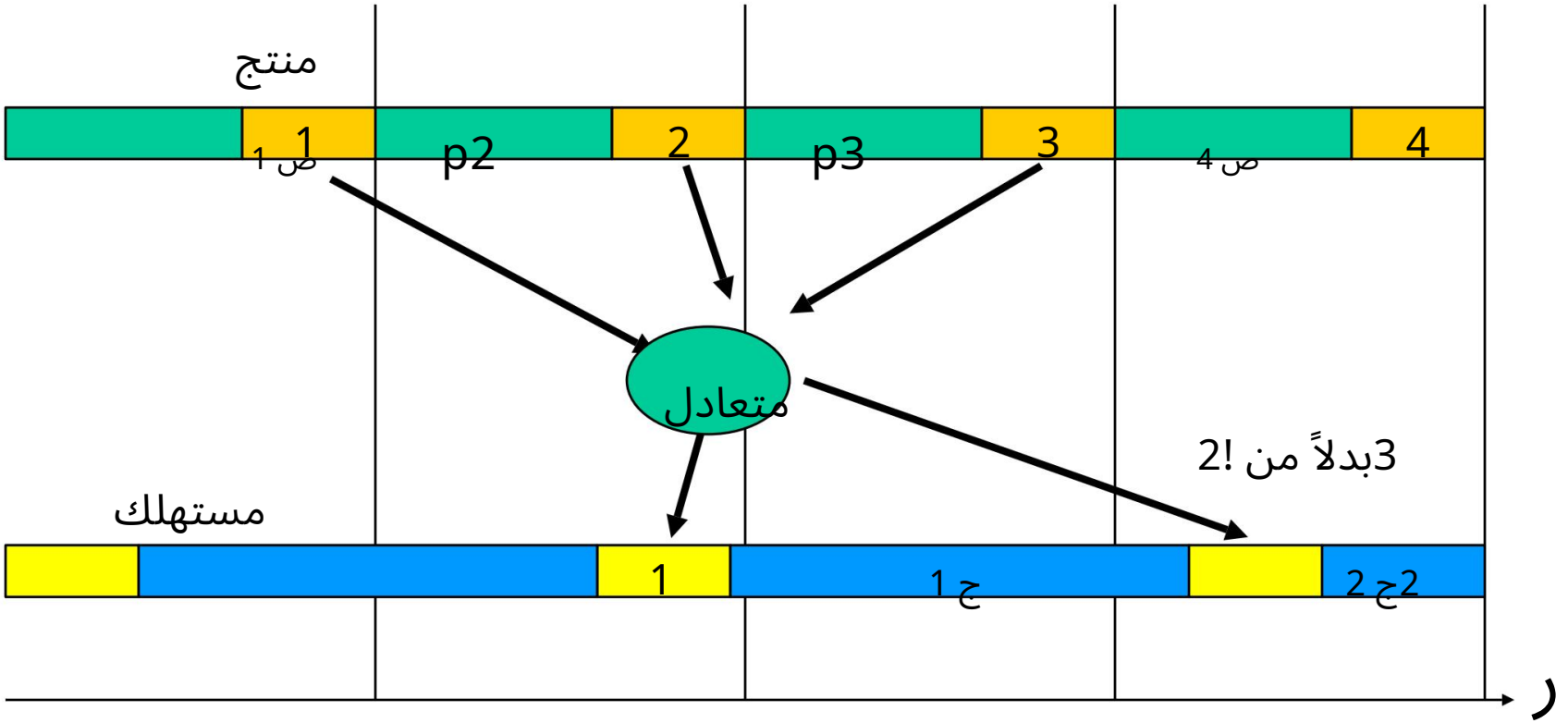
- Interprocess الاتصالات• Deadlocks
- مشاكل IPC الكلاسيكية•

المنتج -مشكلة المستهلك



- المخزن المؤقت مشترك (أي أنه متغير مشترك)

التقدم بمرور الوقت... ..



- يتم بدء كلتا العمليتين في نفس الوقت ويستخدم المستهلك بعض القيمة القديمة في البداية

شرط السباق

- بسبب التوقيت والتي
  تبدأ العملية أولاً
- هناك فرصة مختلفة

قد تنتهي عمليات الإعدام بنتائج مختلفة

الأقسام الحرجة

- جزء حرج

-قسم من الكود يتم فيه العملية
يصل إلى المتغيرات المشتركة ويعدلها

- استبعاد متبادل

-طريقة لمنع لضمان أن واحدة (أو عدد محدد) من العمليات في
قسم حرج

لماذا تحتاج العمليات إلى التواصل؟

- لمزامنة عمليات الإعدام الخاصة بهم
- تبادل البيانات والمعلومات

قواعد لتشكيل الأقسام الحرجة

1. لا يجوز إجراء عمليتين في نفس الوقت داخل CS (الاستبعاد المتبادل)

2. لا توجد افتراضات بشأن سرعات المعالجة النسبية أو عدد وحدات المعالجة المركزية.

3. يجب ألا تمنع العملية خارج CS العمليات الأخرى

4. يجب ألا تنتظر أي عملية إلى الأبد قبل الدخول إلى CS الخاص بها

مشكلة الاستبعاد المتبادل
مجاعة

- تعريف

-التأخير إلى أجل غير مسمى في جدولة عملية لصالح العمليات الأخرى

- موجه

-عادة التحيز في سياسات جدولة الأنظمة (خوارزمية جدولة سيئة) • الحل

-تنفيذ بعض أشكال الشيخوخة

مشكلة أخرى الجمود

-يتم حظر عمليتين (أو أكثر) في انتظار حدث لن يحدث أبدًا

-بشكل عام ، ينتظر Aأن يفعل Bشيئًا وينتظر B A

-كلاهما لا يفعل أي شيء حتى كلا الحدثين
لا تحدث ابدا

كيفية التنفيذ
استبعاد متبادل

- ثلاثة احتمالات

-التطبيق: يقوم المبرمج ببناء طريقة ما في البرنامج

-العتاد: تعليمات خاصة بالأحذية / ث متوفرة لتنفيذ ME

-نظام التشغيل: يوفر بعض الخدمات التي يمكن للمبرمج استخدامها. •تعتمد جميع المخططات على بعض التعليمات البرمجية لـ enter_critical_section -و exit_critical_section -

تطبيق
استبعاد متبادل

- تطبيق الاستبعاد المتبادل

-تم تنفيذه بواسطة المبرمج -يصعب الحصول عليه بشكل صحيح وغير فعال للغاية

- تعتمد جميعها على شكل من أشكال الانتظار المزدحم (تختبر العملية شرطًا ، وتعيين علامة ، وحلقات بينما تظل الحالة كما هي)

مثال

- منتج المنتج

إذا كان القفل = حلقة واحدة حتى القفل 0 =
قفل 1 = ضع في قفل المخزن المؤقت 0 =

- مستهلك

إذا كان القفل = حلقة واحدة حتى القفل 0 =
قفل 1 = احصل على من قفل المخزن المؤقت
0 =

تستهلك

الأجهزة: ME

اختبار وتعيين التعليمات

- نفذ x: = r و r: = 1
- x متغير محلي
- r هو سجل عالمي تم ضبطه على 0 مبدئيًا

- كرر (test-set (x)) حتى x = 0 ؛

<قسم حرج>

ص: ؛ 0 =

الأجهزة :ME

تعليمات الصرف

•التبادل: قم بتبديل قيم xو x • rهو متغير محلي r •هو سجل عالمي تم تعيينه على 1في البداية

•س: ؛ 0 =كرر التبادل (ص ، س) حتى س ؛ 1 =<قسم حرج> الصرف (ص ، خ) ؛

ملاحظة: r: = 0 و x: = 1 عندما تكون العملية في CS

خصائص الأجهزة ME

- المزايا -يمكن استخدامها من خلال عمليات مفردة أو متعددة (مع ذاكرة مشتركة) -بسيطة وبالتالي يسهل التحقق منها -يمكنها دعم أقسام مهمة متعددة

- العيوب -يتم استخدام الانتظار المزدحم -المجاعة ممكنة -المأزق ممكن (خاصة مع الأولويات)

جهاز آخر :ME
تعطيل المقاطعات

•على وحدة معالجة مركزية واحدة يتم تنفيذ عملية واحدة فقط •يتم تحقيق التزامن من خلال تنفيذ التشذير (عادة ما يتم ذلك باستخدام المقاطعات)

•إذا قمت بتعطيل المقاطعات ، فيمكنك التأكد فقط سيتم تنفيذ عملية واحدة على الإطلاق

•عملية واحدة يمكن أن تقفل النظام أو تؤدي إلى تدهور الأداء بشكل كبير

استبعاد متبادل

من خلال نظام التشغيل

- الإشارات. • تمرير الرسالة

إشارات

- تقدم كبير تم دمجه في العديد من أنظمة التشغيل الحديثة (Unix ، OS / 2)

- الإشارة هي -عدد صحيح غير سالب -لها عمليتان صالحتان

عمليات سيمافور

•انتظر (فترات)

إذا كانت ، s> 0فإن s: = s - 1

آخر يمنع هذه العملية •إشارة (إشارات)

إذا كانت هناك عملية محظورة على هذه الإشارة ،
فقم بإيقاظها وإلا s: = s + 1

المزيد عن Semaphores

- نوعان من الإشارات -يمكن أن تكون الإشارات الثنائية 0أو 1 فقط -يمكن أن يكون عد الإشارات غير سالب

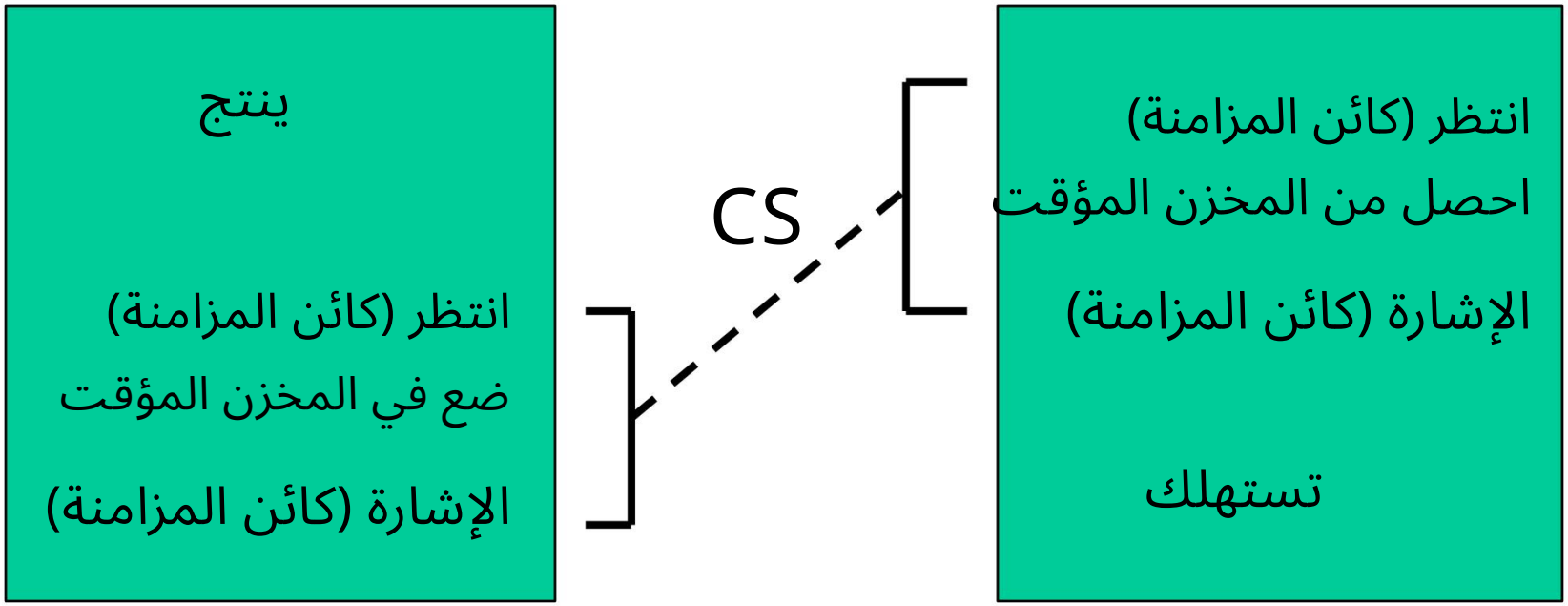
عدد صحيح

- Semaphores هي خدمة نظام تشغيل يتم تنفيذها باستخدام إحدى الطرق الموضحة بالفعل -عادةً عن طريق تعطيل المقاطعات لفترة قصيرة جدًا

الوقت

المنتج -مشكلة المستهلك:

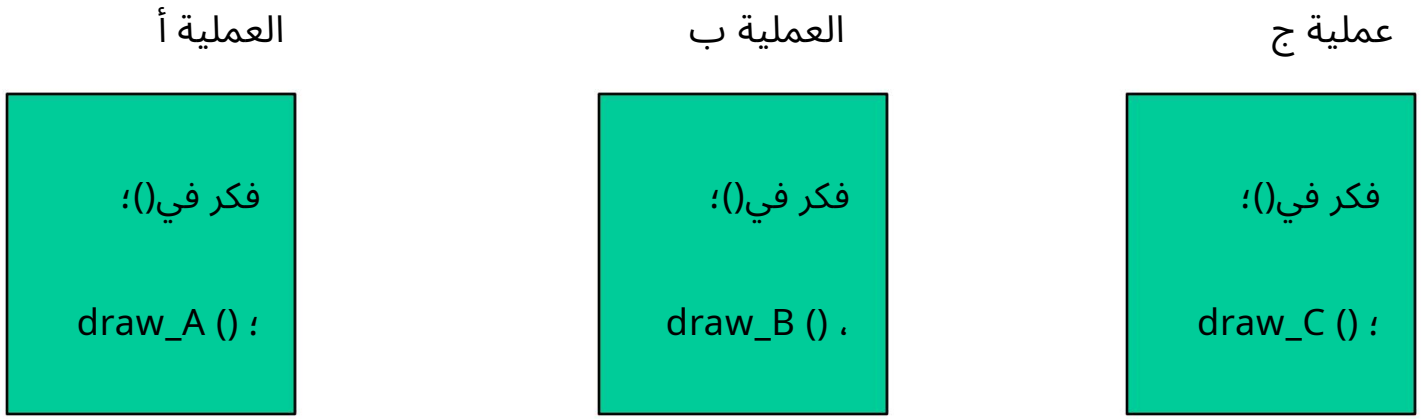
الحل بواسطة Semaphores



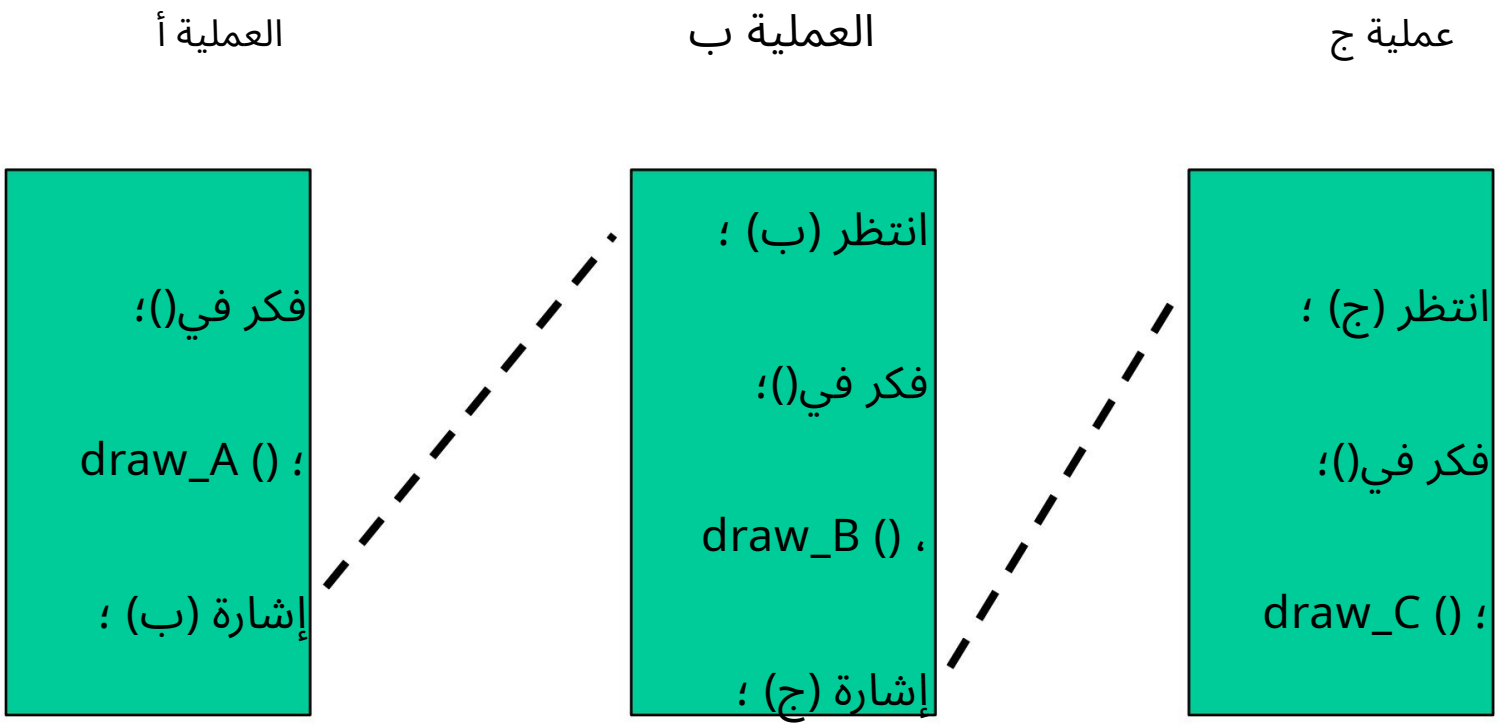
- في البداية إشارة كائن المزامنة هو 1

مثال آخر

- ثلاث عمليات تشترك جميعها في مورد حيث - يرسم المرء الحرف أ
  - واحد يرسم ب
  - واحد يرسم ج
- تنفيذ شكل من أشكال التزامن بحيث يتم إخراجها
  يظهر ABC



•الإشارة ب ، 0 =ج ؛ 0 =



مشكلة العازلة المقيدة

- نحتاج إلى 3 إشارات:

-نحن بحاجة إلى إشارة كائن المزامنة (semaphore) لتكون متبادلة الاستبعاد عند الوصول إلى المخزن المؤقت. -نحن بحاجة

إلى إشارة كاملة لمزامنة المنتج والمستهلك على عدد العناصر الاستهلاكية. -نحن بحاجة إلى إشارة فارغة للمزامنة

المنتج والمستهلك على عدد المساحات الفارغة.

عازلة مقيدة - إشارات

- البيانات المشتركة

إشارة ممتلئة ، فارغة ، كائن المزامنة ؛

في البداية:

ممتلئ ، 0 =فارغ ، n =كائن المزامنة 1 =

عازلة محدودة -عملية المنتج

```
فعل }
...
إنتاج عنصر في nextp
...
انتظر (فارغ) ؛ انتظر
(كائن المزامنة) ؛
...
إضافة nextpإلى المخزن المؤقت
...
إشارة (كائن المزامنة) ؛
إشارة (ممتلئة) ؛ }
بينما ؛ (1)
```

عازلة محدودة -عملية المستهلك

```
فعل }
        انتظر (ممتلئ) انتظر
        (كائن المزامنة) ؛
        ...
    إزالة العنصر من المخزن المؤقت إلى nextc
        ...
    إشارة (كائن
المزامنة) ؛ إشارة (فارغة) ؛
        ...
        تستهلك العنصر في nextc
        ...
{بينما ؛ (1)
```

ملاحظات حول حل المخزن المقيد

•ملاحظات (من وجهة نظر المستهلك):

-إن وضع إشارة (فارغة) داخل CSللمستهلك (بدلاً من الخارج) ليس له أي تأثير حيث يجب على المنتج دائمًا انتظار كلتا الإشارات قبل المتابعة.

-يجب أن يقوم المستهلك بالانتظار (ممتلئ) قبل الانتظار (كائن المزامنة) ، وإلا يحدث طريق مسدود إذا قام المستهلك بإدخال CSبينما يكون المخزن المؤقت فارغًا. •الخلاصة: استخدام الإشارات فن صعب ...

ما هو الجمود؟

- عملية الجمود

-وصلت العملية إلى طريق مسدود عندما تنتظر حدثًا لن يحدث أبدًا • توقف النظام

-وصل النظام إلى طريق مسدود عند وصول عملية واحدة أو أكثر إلى طريق مسدود

الشروط اللازمة ل
طريق مسدود

- استبعاد متبادل

-يتم استخدام الموارد المشتركة بطريقة حصرية

- Hold & Wait للطرفين

-تحتفظ العمليات بالموارد التي لديها بالفعل أثناء انتظار تخصيص الموارد الأخرى

الشروط اللازمة للوصول إلى طريق مسدود (تابع)

- لا يوجد الشفعة

-لا يمكن استباق الموارد حتى تطلقها العملية

- انتظار دائري

-توجد سلسلة دائرية من العمليات تحتوي فيها كل عملية على الموارد المطلوبة من قبل العملية التالية في السلسلة

لا حالة الجمود

إذا تمكنت من منع واحد على الأقل من حالات الجمود الضرورية ، فلن يكون لديك DEADLOCK

خوارزمية النعامة

•التظاهر بعدم وجود مشكلة •معقول إذا -تحدث حالات توقف تام نادرًا جدًا -تكلفة المنع مرتفعة •يتبع نظام UNIX و Windowsهذا النهج •إنها مقايضة بين

-الراحة -الصواب

طرق التعامل مع الجمود

- الوقاية من الطريق المسدود
- كشف الجمود
- تجنب الجمود
- استعادة حالة الجمود

الوقاية من الطريق المسدود

•إزالة إمكانية حدوث مأزق من خلال إنكار أحد الشروط الأربعة الضرورية: -الاستبعاد المتبادل (هل يمكننا مشاركة كل شيء؟)

-انتظر وانتظر

-لا استباق

-إنتظار دائري

إنكار "انتظر وانتظر"

- التنفيذ

-تعطى عملية مواردها على "الكل

أو لا شيء "

-إما أن تحصل العملية على كل ما هو مطلوب

الموارد والعائدات أو يحصل على أي منهم وينتظر حتى يستطيع

إنكار "انتظر وانتظر"

- مزايا
  - إنها تعمل
  - سهولة البرمجة بشكل معقول
- مشاكل
  - هدر الموارد
  - إمكانية الجوع

إنكار "عدم الاستباقية"

- التنفيذ

-عندما يتم رفض عملية طلب مورد ، يجب أن تحرر جميع الموارد الأخرى التي تحتفظ بها

-يمكن إزالة الموارد من ملف العملية قبل أن تنتهي معهم

إنكار "عدم الاستباقية"

- مزايا

-إنها تعمل

-استخدام الموارد بشكل أفضل. • المشاكل

-تكلفة إزالة العملية

مصادر

-من المحتمل أن تفقد العملية العمل الذي قامت به.
(كيف غالبا ما يحدث هذا؟)

-إمكانية الجوع

رفض "انتظار دائري"

- التنفيذ
- الموارد مرقمة بشكل فريد
- يمكن للعمليات طلب الموارد بترتيب تصاعدي خطي فقط
- وبالتالي منع حدوث الانتظار الدائري

رفض "انتظار دائري"

- مزايا
- يعمل
- المشاكل

- يجب طلب الموارد بترتيب تصاعدي لرقم المورد وليس حسب الحاجة

- يجب الحفاظ على ترقيم الموارد من قبل شخص ما ويجب أن يعكس كل إضافة إلى نظام التشغيل

- من الصعب الجلوس وكتابة الكود

تجنب الجمود

- السماح لفرصة حدوث طريق مسدود
- ولكن تجنب حدوث ذلك ..

- تحقق مما إذا كانت الحالة التالية (التغيير في النظام) قد ينتهي بها الأمر في حالة توقف تام

خوارزمية مصرفي

•تعريفات

-كل عملية لها قرض ، مطالبة ،
الحد الأقصى من الحاجة

•القرض: العدد الحالي من الموارد المحتفظ بها

•الحد الأقصى المطلوب: إجمالي موارد العدد المطلوب للإكمال •

المطالبة: =(الحد الأقصى -القرض)

مشكلة مصرفي

عميل	الأعلى. يحتاج	القرض الحالي	مطالبة
1 ج	800	410	390
2 ج	600	210	390

- لنفترض أن إجمالي رأس مال البنك هو 1000دولار أمريكي.
- النقد الحالي: 380 = (210 + 410) -1000

الافتراضات

- إنشاء سقف قرض (الحد الأقصى للحاجة) لكل عملية

-الحد الأقصى المطلوب <إجمالي عدد الموارد المتاحة (مثل رأس المال) • يجب أن يكون إجمالي القروض للعملية أقل من أو يساوي

الحد الأقصى المطلوب • يجب إعادة الموارد التي تم إقراضها في وقت محدد

الخوارزمية

1. ابحث عن عملية بمطالبة يمكن إرضائها باستخدام العدد الحالي من الموارد المتبقية (على سبيل المثال ، منح المطالبة مبدئيًا) .2

إذا تم العثور على مثل هذه العملية ، فافترض أنها ستعيد الموارد المعارة.

3. قم بتحديث عدد الموارد المتبقية .4 كرر الخطوات من 1 إلى 3 لجميع العمليات وقم بتمييزها

الخوارزمية

•لا تمنح المطالبة إذا تعذر وضع علامة على عملية واحدة على الأقل.

•التنفيذ

-لا يُسمح بطلب مورد إلا إذا نتج عنه

في حالة آمنة

-يتم الحفاظ على النظام دائمًا في حالة آمنة ، لذلك سيتم ملء جميع الطلبات في النهاية

الخوارزمية

- مزايا

- يسمح للمهام بالمضي قدمًا عندما لا تعمل خوارزمية الوقاية من المشاكل

- يتطلب أن يكون هناك رقم ثابت من مصادر

- ماذا يحدث إذا تعطل المورد؟

- لا يسمح للعملية بتغيير أقصى حاجتها أثناء المعالجة

مشاكل IPC الكلاسيكية

- مشكلة القراء والكتاب

-نماذج الوصول إلى قاعدة بيانات (قراءة و كتابة)

- تناول الطعام مشكلة الفلاسفة

-عمليات النماذج المتنافسة للوصول الحصري إلى عدد محدود من الموارد مثل أجهزة الإدخال / الإخراج

- مشكلة الحلاق النائم

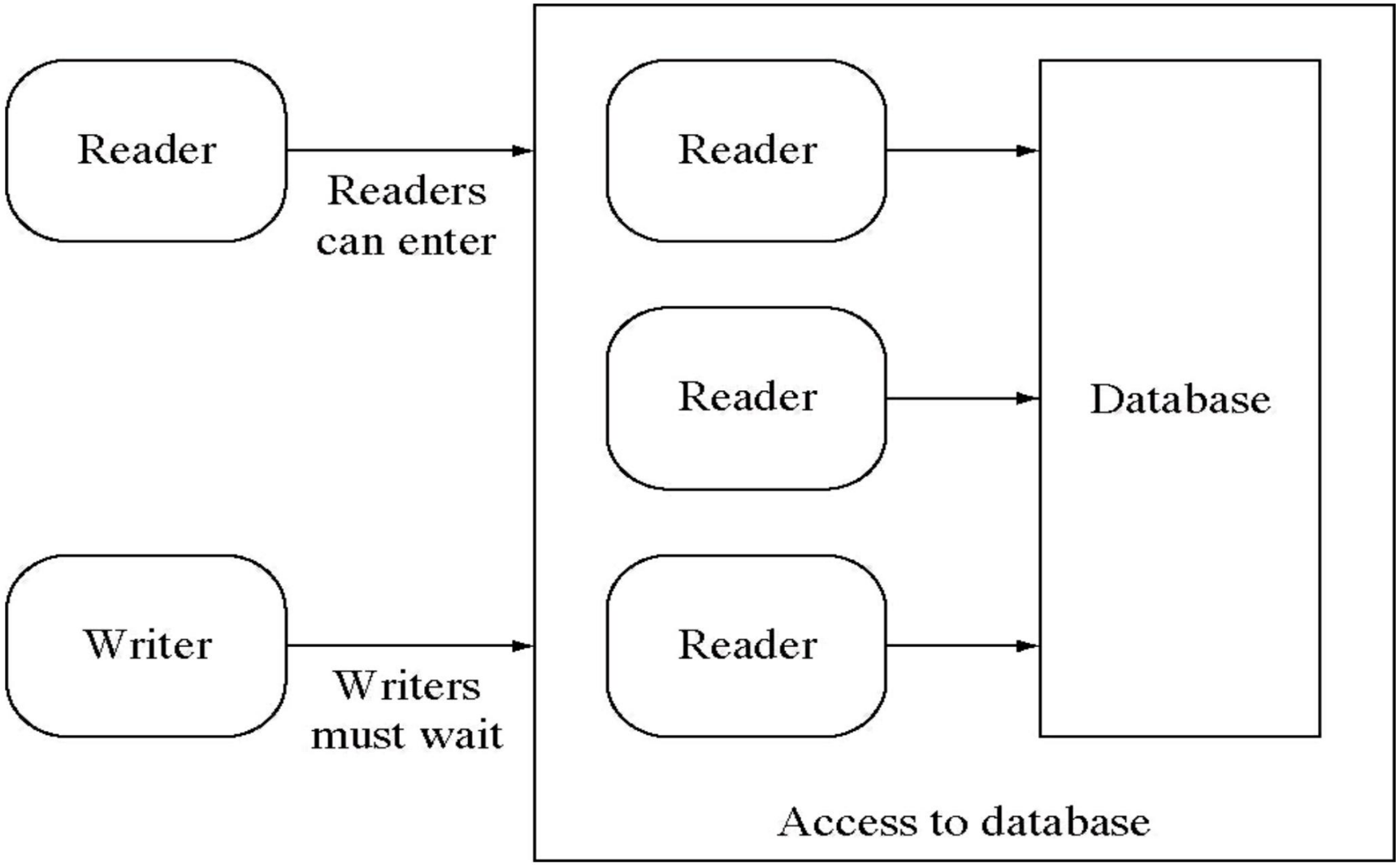
-نماذج مواقف الانتظار مثل مكتب المساعدة متعدد الأشخاص مع نظام انتظار المكالمات المحوسب لعقد عدد محدود من المكالمات الواردة

مشكلة القراء والكتاب

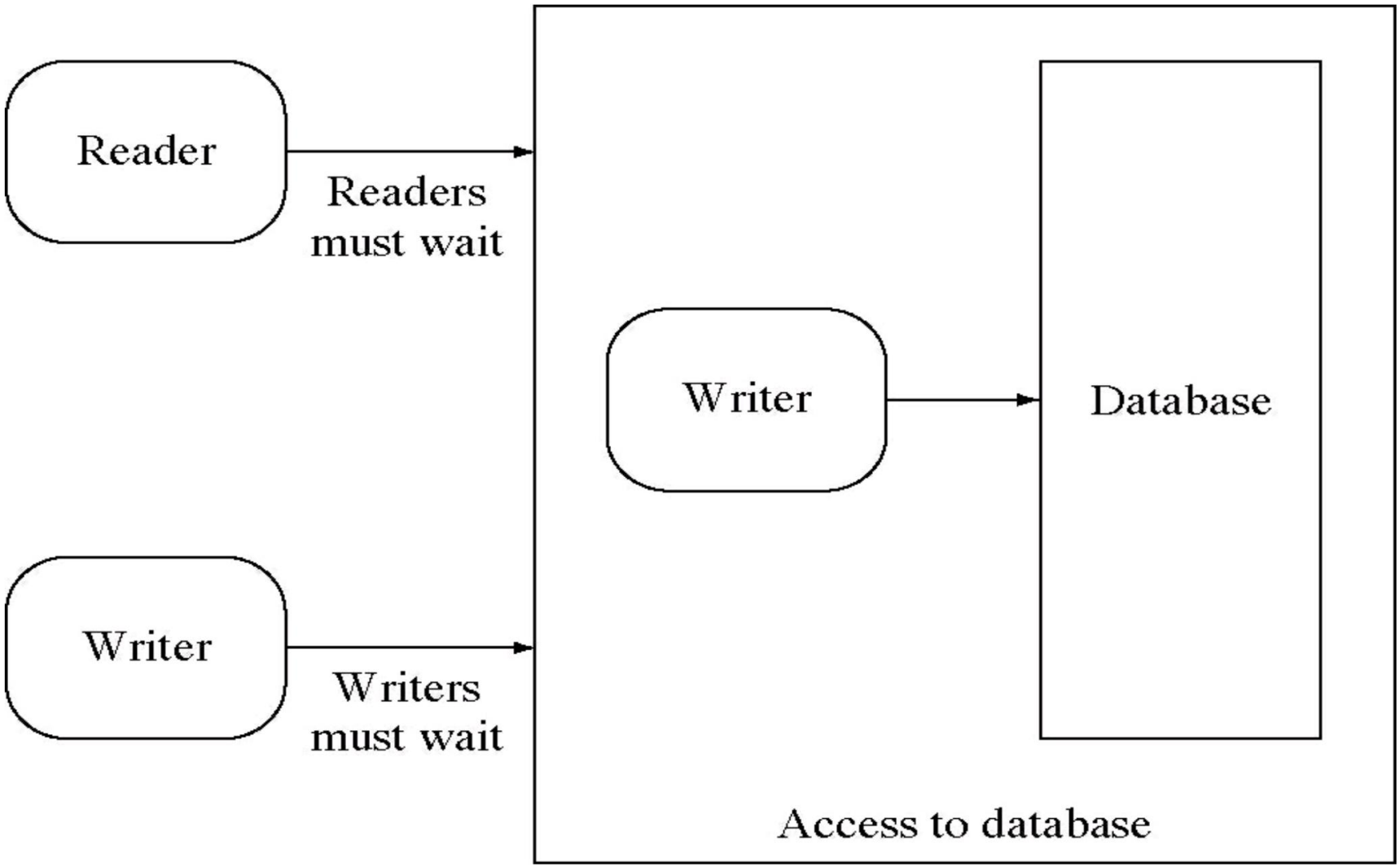
•يتم تشغيل أي عدد من أنشطة القراء وأنشطة الكاتب. •في أي وقت ، قد يرغب نشاط القارئ في قراءة البيانات.

•في أي وقت ، قد يرغب نشاط الكاتب في تعديل البيانات. •يجوز لأي عدد من القراء الوصول إلى البيانات في وقت واحد. •خلال الوقت الذي يكتب فيه الكاتب ، لا يجوز لأي قارئ أو كاتب آخر الوصول إلى البيانات المشتركة.

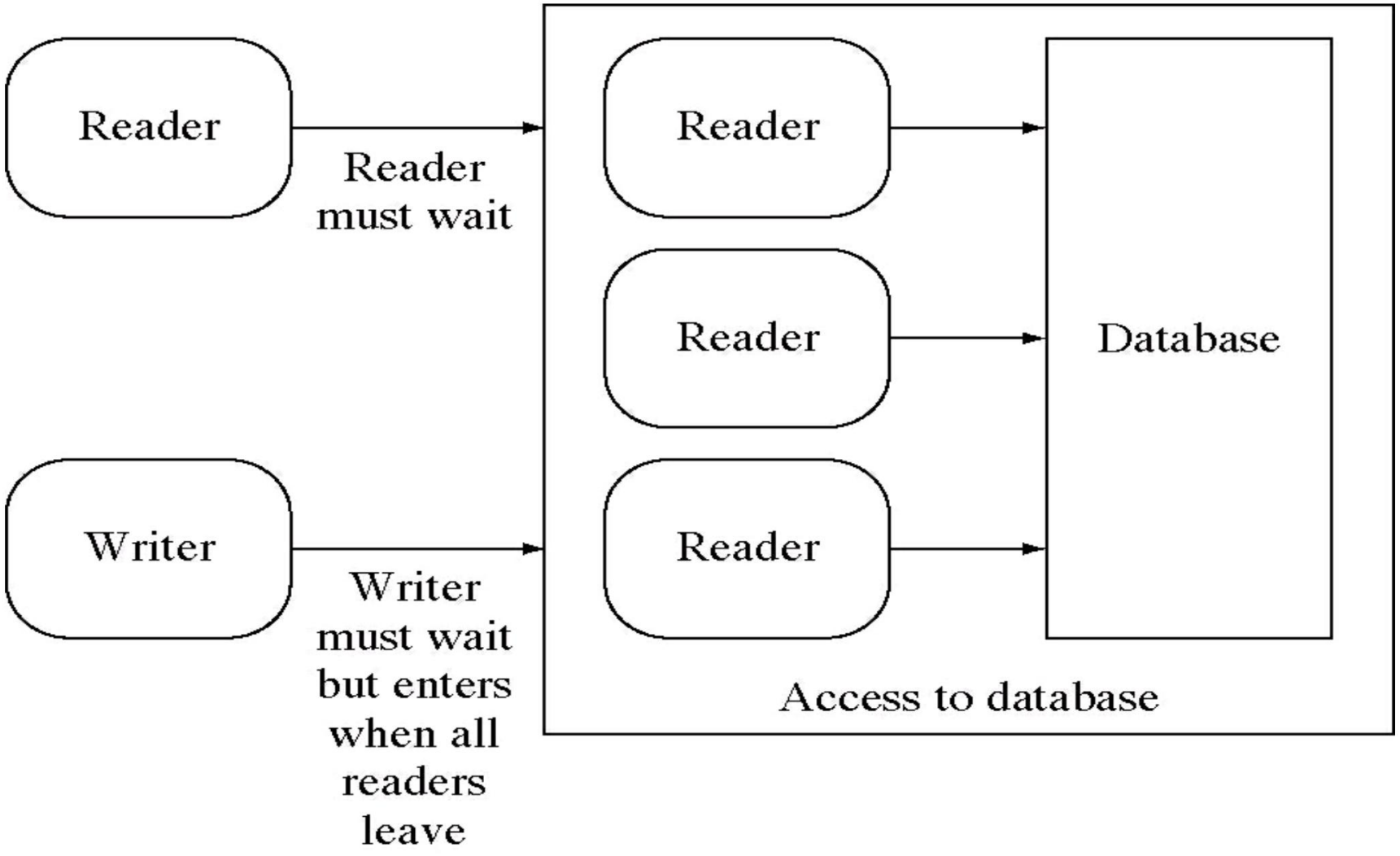
القراء والكتاب مع القراء النشطين



القراء والكتاب مع كاتب نشط



هل يجب أن ينتظر القراء انتظار الكاتب؟



مشكلة القراء والكتاب

•تتطلب مشكلة القراء-الكتاب الأولى عدم بقاء أي قارئ في انتظار ما لم يتمكن الكاتب من الوصول إلى البيانات المشتركة.

•تتطلب مشكلة القراء-الكتاب الثانية أنه بمجرد أن يصبح الكاتب جاهزًا ، لا يجوز للقراء الجدد البدء في القراءة. •في حل الحالة الأولى قد يتضور الكتاب جوعاً ؛ في حل الحالة الثانية قد يموت القراء جوعا.

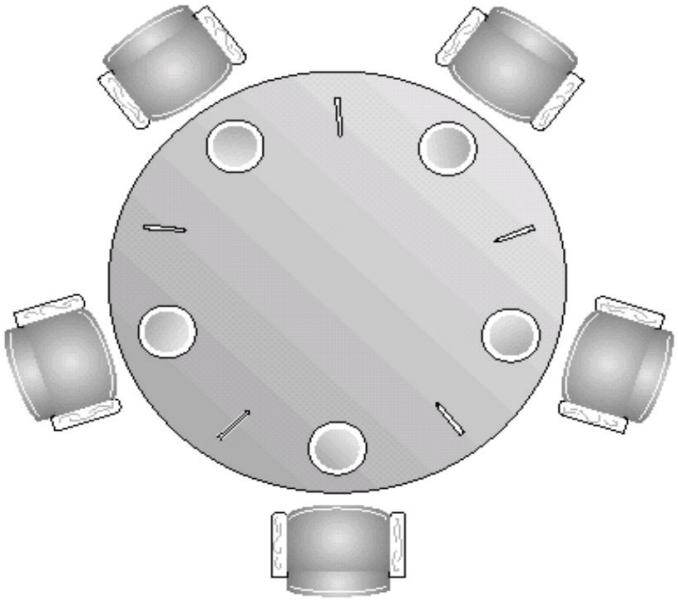
أول حل للقراء والكتاب

- عداد القراءة يتتبع عدد العمليات التي تتم قراءتها حاليًا.

- إشارة كائن المزامنة (mutex semaphore) يوفر الاستثناء المتبادل لتحديث readcount. • wrt semaphore يوفر الاستثناء المتبادل للكتاب ؛ يتم استخدامه أيضًا من قبل القارئ الأول أو الأخير الذي يدخل CS أو يخرج منه.

مشكلة الفلاسفة في تناول الطعام

- •يجلس خمسة فلاسفة حول طاولة دائرية. أمام كل طبق وعاء أرز. •يوجد بين كل زوج من الناس عيدان طعام (شوكة) ، لذلك هناك خمسة أعواد طعام.



•يستغرق تناول الأرز اثنين من عيدان تناول الطعام (شوكة؟) ، لذلك بينما يأكل nلا يمكن أن يأكل n + 1أو .n-1

مشكلة الفلاسفة في تناول الطعام

- يفكر كل شخص لفترة ، ويحصل على عيدان تناول الطعام اللازمة ، ويأكل ، ويضع عيدان تناول الطعام مرة أخرى ، في دورة لا نهاية لها. • يوضح صعوبة تخصيص الموارد بين العملية دون الجمود والمجاعة.

مشكلة الفلاسفة في تناول الطعام

- التحدي هو تلبية طلبات عيدان تناول الطعام مع تجنب المأزق والمجاعة.

- يمكن أن يحدث الجمود إذا حاول الجميع الحصول على عيدان تناول الطعام مرة واحدة. يحصل كل منهما على عود طعام يسار ، ويكون عالقًا ، لأن كل عود طعام يمين هو عصا طعام يسرى لشخص آخر.

حل الفلاسفة لتناول الطعام

العملية :Piكرر التفكير ؛ انتظر (شوكة ؛ ([i]انتظر
؛ ([fork [i + 1 mod 5)تأكل؛ إشارة (شوكة ؛ ([5
i + 1 mod]إشارة (شوكة ؛ ([i]إلى الأبد

- كل فيلسوف هو عملية.
- إشارة واحدة لكل

مفترق: -مفترق: صفيف
[0..4]من الإشارات

-التهيئة: fork [i] .count: = 1 for i: = 0..4

المحاولة الأولى: طريق مسدود إذا بدأ كل فيلسوف برفع شوكته اليسرى (عود)!

حل الفلاسفة لتناول الطعام

الحلول الممكنة لتجنب المأزق: •السماح لأربعة فلاسفة على

الأكثر بالجلوس على الطاولة في نفس الوقت. •يقوم الفيلسوف ذو الأرقام الفردية بالتقاط الشوكة اليسرى أولاً ، حتى أنه يلتقط الشوكة اليمنى.

ضعف السيمافور

-من المتوقع أن يكتب المستخدم الانتظار والإشارة بالترتيب الصحيح.

-يجب على المستخدم أن يتذكر تنفيذ إشارة لكل منها

خروج

-قد تنتشر المكالمات في جميع أنحاء البرنامج -قد يتطلب المنطق أن تقوم العملية بفحص أقرانه والإشارة إليهم. •راجع منطق حل • .Philosophers Diningكان هناك عدد كبير من البدائل المقترحة. •الشاشات هي نهج واحد شائع

الشاشات

•بناء لغة جديد يتضمن المزامنة •تم تنفيذه في Java عبر كلمة أساسية متزامنة •يمكن لعملية واحدة فقط إدخال مراقب (استخدام نقاط الإدخال) في كل مرة. •على سبيل المثال ، قد يقوم المترجم بإنشاء استدعاء لإشارة مشتركة عند الإدخال: إشارة عند الخروج. •إذا قامت العملية P بإجراء مكالمة للشاشة ، وكانت العملية Q في الشاشة ، فسيتم حظر P حتى خروج Q

مثال جافا

```
حساب فئة {رصيد مزدوج خاص؛ الحساب العام (إيداع
مزدوج) {}الرصيد =الإيداع ؛ getBalanceالمزدوج
المتزامن العام ()}رصيد الإرجاع ؛

{إيداع عام متزامن باطل (مبلغ مضاعف) {الرصيد = +المبلغ ؛

{سحب عام باطل متزامن (مبلغ مضاعف) ؛amount = -
{Balance

}
}
```

التكافؤ

- الشاشات والإشارات لها ما يعادلها قوة

- أي شيء يمكنك القيام به مع الشاشات ، يمكنك القيام به مع الإشارات.

- أي شيء يمكنك القيام به إشارة. - إثبات: يمكننا تنفيذ شاشة بإشارة.
- Proof: يمكنك القيام به باستخدام Semaphores ، باستخدام Monitors يمكننا تنفيذ إشارة باستخدام علامة

مراقب

ملخص

•لقد رأينا مشكلتين

-أقسام حرجة -لا يمكن تعديل كليهما

عامل

-التزامن -يجب تحديد الترتيب •غالبًا ما تكون مشاكلنا مزيجًا من الاثنين

-القراء / الكتاب يتشاركون التخزين ، وعلى القراء الانتظار إذا كان هناك كتاب ينتظرون. •من الصعب استخدام الإشارات بشكل صحيح. •بينما كان هناك دعم لغوي للشاشات لبعض الوقت ، لا يزال نظام UNIXالقياسي يدعم فقط semaphore (man sem_open)