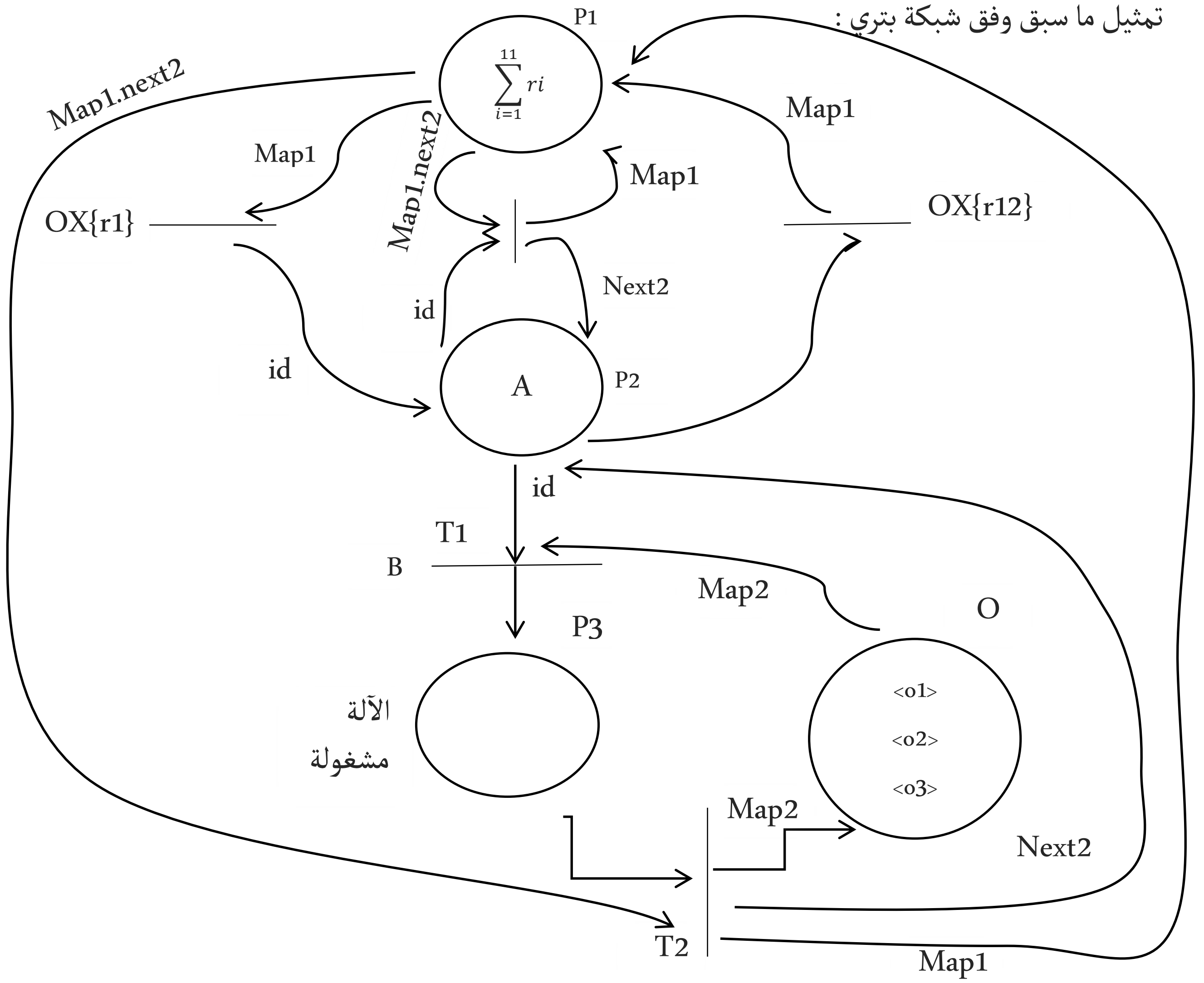
سنكمل في هذه المحاضرات مسألة (*FIFO*) المذكورة في المحاضرة السابقة مع إضافة بعض التعديلات والشروط عليها، بفرض لدينا *12* حجرة (بدلاً من *8* حجرات) ، إضافة إلى الشروط :

1. القطعة الأولى تخرج من الحجرة الثالثة وتعالج في الآلة الأولى *<o1,r3>*.
2. القطعة الثانية *<o2,r7>* تخرج من الحجرة السابعة وتعالج في الآلة الثانية.
3. القطعة الثالثة *<o3,r10>* تخرج من الحجرة العاشرة وتعالج في الآلة الثالثة.

تمثيل ما سبق وفق شبكة بتري :



**ملاحظة:** العبور يكون وفقاً للون الممر وليس وفقاً للون المكان.

- **توضيحات:**

*OX{r1}*: جميع الأغراض يجب أن تدخل من الحجرة *<r1>*.

*OX{r12}*: جميع الأغراض يجب أن تخرج من الحجرة *<r12>*.

*Map1*: تابع يقوم بإلغاء المركبة الأولى، بفرض إذا كان الدخل *<o1,r1>* سيرد *<r1>* فقط.

*Map2*: تابع يقوم بإلغاء المركبة الثانية، بفرض إذا كان الدخل *<o1,r1>* سيرد *<o1>* فقط.

*Next2*: تابع يقوم بتغيير قيمة المركبة الثانية بالقيمة التي تليها، على فرض *<o1,r1>* ستصبح *<o1,r2>*.

*Map1.next2*: تابع يقوم بإلغاء المركبة الأولى، ويقوم بتغيير قيمة المركبة الثانية بالقيمة التي تليها، بفرض كان الدخل *<o1,r1>* سيرد *<r2>* وذلك لتحقق من كون الحجرة الثانية فارغة.

*B*: ألوان الممر {<o1,r3>,<o2,r7>,<o3,r10>}

*A={<r1>..<r11>}-B*: يسمح بانتقال جميع الأغراض ماعدا في الحجرة *<r12>* ستخرج من النظام و الحجرات *{<r3>,<r7>,<r10>}* ستسلك سلوك آخر (الشيء المضاف عن المحاضرة السابقة).

- **السيناريو:**

لعبور الممر *T1* يلزمنا تحقيق لون من ألوان الممر B وهي (*<r3>* أو *<r7>* أو *<r10>*) تأتي من المكان *A*، (*<O1>* أو *<O2>* أو *<O3>*) تأتي من المكان *O*.

بعد عبور الممر *T1* ومعالجة القطعة حسب لون الممر الموافق، بعد انتهاء المعالجة يلزمنا التحقق من أن الحجرة التالية للحجرة الحالية فارغة، نطبق *map1.next2* على القوس الآتي من المكان *P1* نعبر الممر *T2* وفقاً للون *B* ويكون الخرج كالتالي:

*Map2*: تحرير الآلة المشغولة.

*Map1*: تحرير الحجرة المشغولة.

*Next2*: حجز *الحجرة* التالية.

- **إضاءة:**

← زيادة الشروط يقابلها إضافة ألوان لنعبر بهذه الشروط.

← المثال السابق يمثل أقصى ما يمكن فعله في شبكات بتري ولا يعد ذلك من الأمر السهل وإنما معقد نوعاً ما.

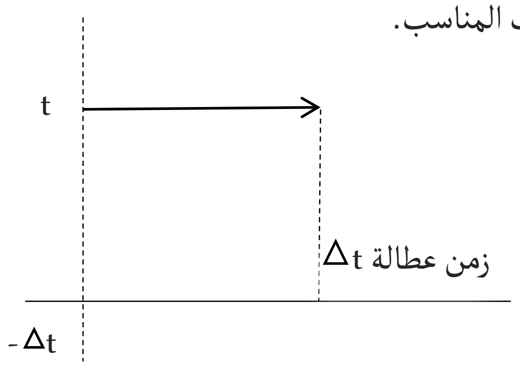
← قمنا بالاستفادة من شبكات بتري في حالات التفريع بالبرنامج عوضاً عن كتابة كود برمجي.

← سلوك شبكة بتري لم يكن عبثاً وإنما السلوك نجري عليه تحقيق لننتقل بعدها إلى كود حقيقي أو تفسير (قيادة نظام مباشرة).

## نظام زمن حقيقي

نظام زمن حقيقي: هو نظام نعتمد صحته ليس فقط على النتائج وإنما على الزمن الذي تم فيه إنجاز العمليات.

لا يهم فيما إذا كان يؤدي وظائفه بشكل صحيح وإنما المهم أن تكون في الوقت المناسب.




**مثلاً:** إصدار أمر في اللحظة t ولكن هذا الأمر لن ينفذ في اللحظة t

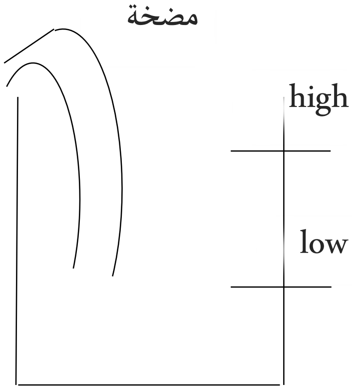
إنما سيأخذ زمن عطالة لذلك أنا بحاجة لإصدار الأمر قبل زمن قدره t

(زمن عطالة) أي تبدأ المهام في وقت أبكر لتنفيذ في الوقت المناسب

(أخذ عطالة المشغلات بعين الاعتبار كما أسلفنا سابقاً).

**مثال:** مناجم الفحم

عندما يقوم العمال بحفر باطن الأرض تخرج المياه ويخرج أيضاً غاز الميثان وغاز أول أكسيد الكربون الذي إذا ازدادت نسبته عن حد معين يسبب الاختناق وإذا تجاوزت نسبة غاز الميثان حد معين




فإنها مع الشرارات (التي تتولد أثناء عملية الحفر)، تؤدي إلى انفجار، تعمل مضخة للتخلص من المياه والتي تعمل عند زيادة منسوب المياه للحد high تقوم بسحب المياه إلى أن يصل منسوب المياه للحد low وتقف، ولكن أيضاً إذا كانت نسبة غاز الميثان فوق حد معين فأنا لا أستطيع تشغيل هذه المضخة لأن ذلك سيولد انفجار فهذا النظام نظام زمن حقيقي حيث يجري مراعاة حساسي المياه والغاز.

**ملاحظة:**

إذا كانت النظم غير خطرة (حساسة) أقوم بعمل نظام مقاطعات بحيث أقوم بإرسال مقاطعات للنظام بحيث إن تأخرت إحدى المهام لن يؤدي ذلك إلى خلل في العمل على عكس الأنظمة الحرجة التي تحتاج لمراقبة دورية.

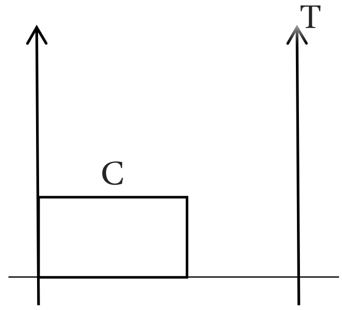
✓ في نظام الزمن الحقيقي:

- مجموعة من الفعاليات (process) متسايرة: عدة فعاليات تعمل على شكل threads (فعالية لتشغيل مضخة، فعالية لإيقافها، فعالية تقيس غاز الميثان ...).
- لدينا متطلبات زمنية.
- إشارات رقيمية (تحويل من Analog إلى digital وبالعكس): لكي نأخذ بعين الاعتبار وجود عطالات.
- التنفيذ يعتمد على الزمن: حيث يجب الالتزام بالمحددات الزمنية وعدم تجاوزها.
- فحص هذه الأنظمة ليس بالسهل: بسبب وجود model رياضي معقد للتعامل معها.
- حرجة من ناحية الأمان: لأنها غالباً تتعامل مع الأنظمة الحرجة.
- نهتم فيها بالزمن الأسوء (ليس المتوسط): حيث نحسب دوماً من أجل الحالة الأسوأ لأن الحالة المتوسطة يمكن ألا تراعي الحالة الأسوأ.
- شديدة الترابط بال Hardware (بالوسط المحيط).
- هي أنظمة حتمية، موثوقة وتكهنية.

✓ تكهنية (تنبؤية): أي نستطيع التنبؤ بالدور T (زمن المهمة)

بداية لمعرفة كيف يتم وضع دور المهام:

مثلاً المهمة التي تقوم بامتصاص غاز الميثان تقوم بالعمل كل فترة محددة (دور) أحدده من خلال حساب الوقت اللازم لانتشار هذا الغاز (بأكبر غزارة ممكنة له) إلى أن يصل إلى حالة الانفجار فعندها يجب أن يكون دور المهمة أقل من ذلك الوقت مع مراعاة زمن العطالة (أي أقوم بامتصاص الغاز قبل أن يقترب من حالة الخطر لكي أبقى في الوضع الآمن).




زمن المهمة نستطيع التنبؤ به (بحث التعقيد بالخوارزميات هي عمله).

المهمة هي مهمة نظام زمن حقيقي إذا كان زمنها أكبر من الصفر وأصغر من الدور T

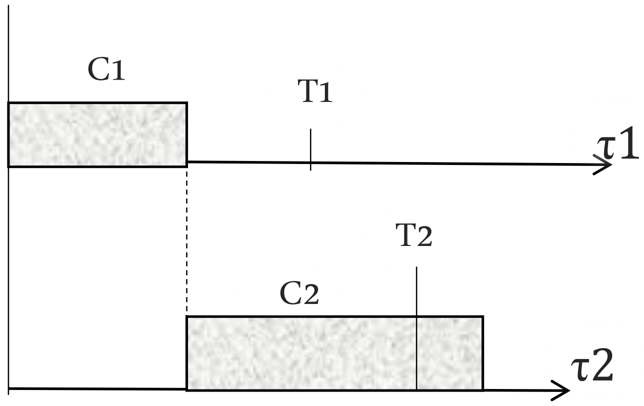
$$0 \leq C < T - D$$

D: زمن عطالة (محرك أو حساس)

T: زمن المهمة يعتمد على سرعة المعالج وتعقيد الخوارزمية

وكل ذلك في حال كانت هذه المهمة هي المهمة الوحيدة في النظام، أما في حال كانت هناك عدة مهام فالأمر مختلف (سيكون أكثر تعقيداً) حيث سيكون أكثر من عملية قياس للحساسات (المهام) والتي هي:

- **المهام التسايرية:**




ليكن لدينا المهام التسايرية التالية:

$\tau2\tau1$, عندما تعمل $\tau1$ لا تعمل $\tau2$

لأن أنظمة الزمن الحقيقي هي أنظمة *primitive schedule*

(أي في حال ورود مهمة ذات أولوية أعلى ستقاطع المهمة ذات الأولوية الأدنى التي تعمل على المعالج وستعمل هي بدلاً منها).

الشرط الأساسي: النظام هو نظام زمن حقيقي عندما تحقق كل فعالياته محدداتها الزمنية.

الشرط المخصص: النظام هو نظام زمن حقيقي إذا كانت كل فعالياته قد حققت موعدها الأول.

في حال تجاوزت المهمة $\tau2$ الزمن $T2$ (كما هو موضح) يكون النظام ليس نظام زمن حقيقي لأن المهمة $\tau2$ فشلت في تحقيق موعدها الأول.

إذا تجاوزت جميع الفعاليات موعدها الأول عندها لن يكون هناك أي خوف على النظام (الموعد الأول يكفي لأنه إن لم يحدث تأخير به لن يتراكم هذا التأخير) وإذا لم يتحقق ذلك أعمل على تصحيحه من خلال:

إما توسيع الدور $T$ (يكون النظام يسمح بالتراخي زمنياً)

أو بتقليل الـ $C$ (عن طريق استخدام معالجات أسرع)

إذا لم أتمكن من توسيع الدور أو تقليل الـ$C$، أتوجه إلى المعالجة التفرعية (أوزع المهام على عدة معالجات مع مراعاة زمن الاتصال).

✓ موثوقة: أي لا تخالف المحددات الزمنية.

✓ حتمية: يتوجب تحقيق الأهداف ضمن الزمن اللازم (لا مجال لنسيان أي شيء بدون حساب)

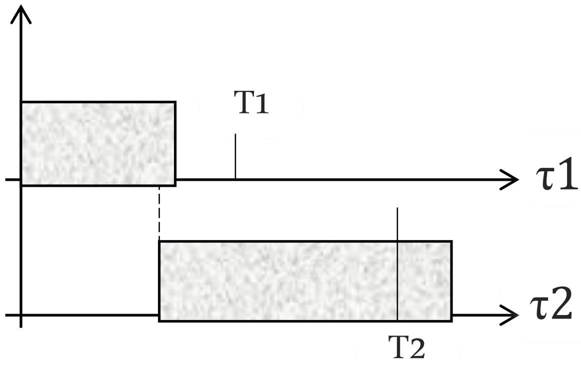
## المعادلات الرياضية لنظام زمن حقيقي:

الشروط اللازمة ليكون النظام نظام زمن حقيقي:

➢ <u>الشرط الأول: (لازم غير كاف)</u>

يجب أن يكون زمن حساب كل مهمة أصغر من دورها $Ci \leq Ti$

مثال: ليكن لدينا المهمتين $\tau2, \tau1$




| | P | T | C |
|---|---|---|---|
| $\tau1$ | 1 | 5 | 3 |
| $\tau2$ | 2 | 10 | 8 |

حيث $p$ هي *priority* (أولوية المهمة)

لنرى هل يتحقق هذا النظام الشرط ؟

$$3 < 5 \leftarrow C1 \leq T1$$
$$8 < 10 \leftarrow C2 \leq T2$$

فالشرط محقق ولكن!

المهمة $\tau1$ أعلى أولوية من $\tau2$ تبدأ هي ثم تنهي المهمة بزمنها ومقداره (3 وحدات زمنية) ثم بعد ذلك تبدأ المهمة $\tau2$ تستلزم زمن قدره (8 وحدات زمنية) مما أدى إلى تجاوز لدورها *T2* (وهو *10* وحدات زمنية) بزمن قدره وحدة زمنية <u>فالنظام ليس نظام زمن حقيقي</u> حيث فشلت $\tau2$ في تحقيق موعدها الأول في حين حققته $\tau1$ (مما يدل على أن الشرط غير كاف).

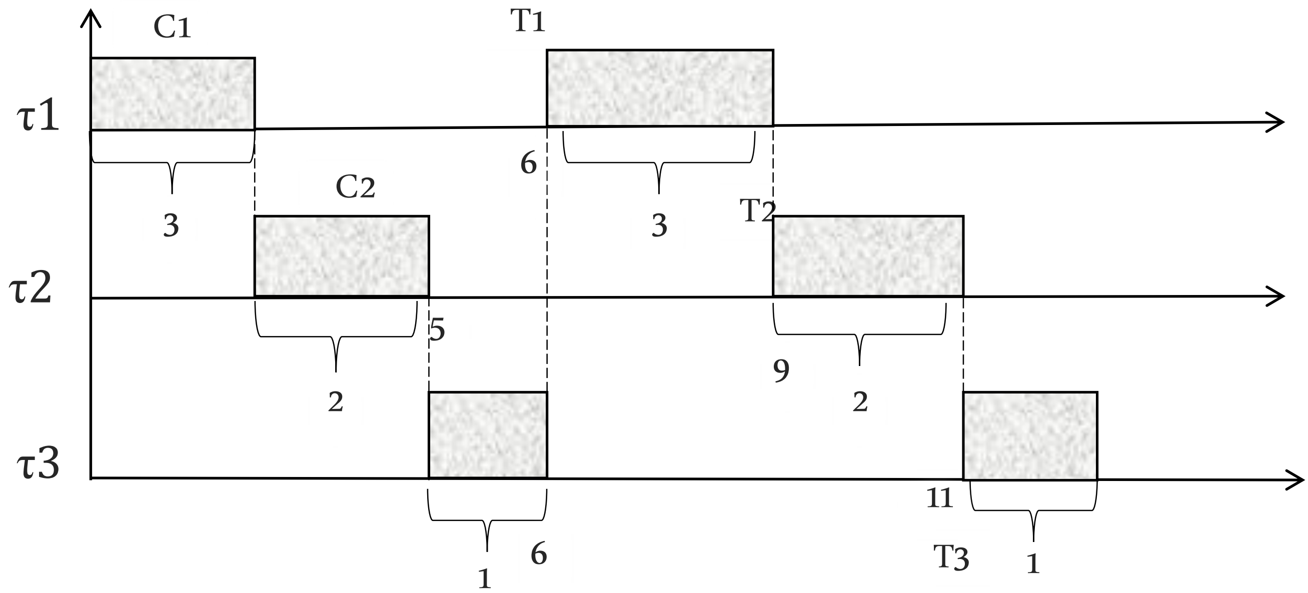
➢ <u>الشرط الثاني: (لازم غير كاف)</u>

$$\text{وهو} \quad \sum \frac{Ci}{Ti} \leq 1$$

حيث: $\sum \frac{Ci}{Ti}$ تسمى الحمولة الوسطية للنظام.

مثال: ليكن لدينا المهام $\tau1, \tau2, \tau3$

| | P | T | C |
|---|---|---|---|
| $\tau1$ | 1 | 6 | 3 |
| $\tau2$ | 2 | 9 | 2 |
| $\tau3$ | 3 | 11 | 2 |




نتأكد من تحقق الشرط حيث: $\frac{3}{6}+\frac{2}{9}+\frac{2}{11}\leq 1$

وهي تقريباً $0.9 < 1$ فالشرط محقق ولكن!

المهمة $\tau 1$ الأعلى أولوية تبدأ هي التنفيذ ثم تنهي المهمة بزمنها وقدره (3 وحدات زمنية) ثم تليها بالأولوية المهمة $\tau 2$ فتتنفذ وتنهي عملها بزمنها وقدره (وحدتين زمنيتين) ثم تليها المهمة $\tau 3$ فتتنفذ ولكن قبل أن تنهي تنفيذها يكون قد بدأ دور جديد للمهمة $\tau 1$ وبما أن $\tau 1$ أعلى أولوية من $\tau 3$ فتقاطع عملها وتنفذ المهمة $\tau 1$ وعندما تنتهي يبدأ دور جديد للمهمة $\tau 2$ ولكن أيضاً $\tau 2$أعلى أولوية من $\tau 3$ فتنفذ $\tau 2$ إلى أن تنتهي يكون قد بدأ دور جديد للمهمة $\tau 3$ ولكن لم تكمل عملها السابق بعد فتكمله بزمن مقداره (وحدة زمنية واحدة) متبقية لها لكي تنهي تنفيذها وبعد ذلك تبدأ دورها الجديد أي أدى ذلك إلى تأخير المهمة $\tau 3$ بمقدار وحدة زمنية واحدة <u>فالنظام ليس نظام زمن حقيقي</u> مع أن المهمتين $\tau 1$, $\tau 2$, حققتا موعدهما الأول إلا أن المهمة $\tau 3$ فشلت في تحقيق موعدها الأول (مما يدل على أن الشرط غير كاف).

- **<u>الشرط الثالث: (لازم غير كاف)</u>**

$$\sum_{j=1}^{i-1}\left(\frac{\mathrm{Ti}}{\mathrm{Tj}}\times \mathrm{Cj}\right)\leq \mathrm{Ti}-\mathrm{Ci}$$

حيث j = 1 ... i-1 هي الفعاليات ذات الأولوية الأعلى من i

$\frac{\mathrm{Ti}}{\mathrm{Tj}}$: كم مرة يمكن للفعاليات أن تقطع الفعالية

فمثلاً في المثال السابق :

$$\frac{T2}{T1}=\frac{9}{6}=1.5=2$$

(2) أتعامل مع أعداد صحيحة فأخذ ما بعد الفاصلة وأعتبره مرة.

أي يمكن أن تقطع $\tau 1$ المهمة $\tau 2$ مرتين خلال دورها.

زمن المهمة $\tau2 = 2$ ودورها=9 (أي لديها مجال تأخير بمقدار T2 - C2= 7) وزمن تنفيذ المهمة $\tau1 = 3$ إذا قطعت $\tau1$ المهمة $\tau2$ مرتين يكون زمن التأخير $\frac{T2}{T1} \times C1 = 2 \times 3 = 6$ أي أقل من مجال التأخير المسموح به لـ $\tau2$ مما يعني أن ذلك لن يؤثر على النظام في حال كان يحوي فقط الفعاليتين $\tau2$ و $\tau1$ ولكن بوجود $\tau3$ يتغير الأمر، فزمن تنفيذ $\tau3 = 2$ ودورها T3=11 (أي لديها مجال للتأخير بمقدار 9 وحدات زمنية).

- $2 \cong \frac{11}{6} = \frac{T3}{T1} \Leftarrow$ تقطع المهمة $\tau1$ المهمة $\tau3$ مرتين خلال دورها أي تأخرها بمقدار

$\frac{T3}{T1} \times C1 = 2 \times 3 = 6$.

- $2 = 1.2 = \frac{11}{9} = \frac{T3}{T2} \Leftarrow$ تقطع المهمة $\tau2$ المهمة $\tau3$ مرتين خلال دورها أي تأخرها بمقدار

$\frac{T3}{T2} \times C2 = 2 \times 2 = 4$.

مقدار التأخير الذي يمكن أن تأخره $\tau2\tau1$ , للمهمة $\tau3$ هو:

$$\sum_{j=1}^{2}\left(\frac{T3}{Tj} \times Cj\right) = 6 + 4 = 10$$

$$10 > 9 \rightarrow T3 - C3$$

الشرط غير محقق والنظام ليس نظام زمن حقيقي.

➢ **الشرط الكافي الغير لازم:**

$$n(2^{1/n}-1) \geq \sum_{i=1}^{n} \frac{Ci}{Ti}$$

حيث n هو عدد الفعاليات، أي إذا كان مثلاً عدد الفعاليات 3 ففي حال كان $3(2^{1/3} - 1)$ أكبر أو تساوي الحمولة الوسطية للنظام عندها يكون النظام نظام زمن حقيقي.

➢ **الشرط الأكيد:**

$$Ri^{n+1} = Ci + \sum_{j \in hp} \left\lceil \frac{Ri^{n}}{Tj} \right\rceil \times Cj$$

زمن الاستجابة الحرج          زمن الحسابات

أي زمن الاستجابة الحرج لـ i = زمن حسابات i + المقاطعات التي تأتي من المهام j ذات الـ high priority أولوية أعلى.

$Ri^{n}$: هو الزمن الحرج في المرحلة السابقة.

زمن الاستجابة الحرج هو زمن الإنهاء الفعلي

وهذا الشرط في حالة تغير مستمر (إضافة حدود إزالة حدود منه) وهو في حال كانت مهام النظام مستقلة عن بعضها (لا يوجد بينها اعتمادية) وهي الحالة المثالية أن تكون المهام لا تتواصل مع بعضها البعض ولكن في الواقع لابد للمهام أن تتواصل فيما بينها.

هذه المعادلة تسمى المعادلة التراجعية لأنه يوجد Ri في طرفي المعادلة.

مثال:

لدينا المهام $\tau1, \tau2, \tau3$ ، نريد التحقق فيما إذا كانت الفعالية الرابعة تحقق محدداتها الزمنية.

ملاحظة:

| | P | T | C |
|---|---|---|---|
| $\tau1$ | 1 | 10 | 1 |
| $\tau2$ | 2 | 12 | 2 |
| $\tau3$ | 3 | 30 | 8 |
| $\tau4$ | 4 | 60 | 20 |

في الحالة العامة لنتحقق من أن النظام نظام زمن حقيقي يجب أن نتحقق بدايةً من المهمة $\tau2$ في حال كانت تحقق محدداتها الزمنية نتحقق من $\tau3$ وفي حال كانت تحقق محدداتها الزمنية نتحقق من $4\tau$ .

نلاحظ أننا لم نتحقق من $\tau1$ وذلك لأنها ذات الأولوية الأعلى وزمن حسابها أصغر من الزمن الكلي فهي لا تتأثر بمهمة سابقة لتفقد محدداتها الزمنية.

لكن هنا سنتحقق من أجل $4\tau$ في المثال لأن $2\tau$ لا تتأثر بـ $1\tau$ لتفقد محدداتها الزمنية والـ $\tau3$ درجة تأثرها بالـ $1\tau$ و $2\tau$ ضمن المجال المسموح به فنرى $4\tau$ لأنه الأعقد والأصعب حيث ستأثر بها $\tau1, \tau2, \tau3$.

$$R(0,20) = input1([0,20[\,) \times 1 + input2([0,20[\,) \times 2 + input3([0,20[\,) \times 8$$

الـ20 هو زمن المهمة $\tau 4$ (C4) (أي أحسب من أجل زمن المهمة وليس دورها)

C1=1

C2=2

C3=8

$input1([0,20[\,)$: أي كم مرة ممكن أن تأتي الفعالية $\tau 1$ ضمن هذا المجال ستأتي مرة من أجل اللحظة 10 (دورها *10*)، والمجال عند الـ20 مفتوح لذلك لا أحسبه مرة ثانية عند 20 أي ستأتي الفعالية $\tau 1$ مرة ضمن هذا المجال.

وبنفس الطريقة نجد أن $input2([0,20[\,) = 2$ و $input3([0,20[\,) = 1$.

$$R(0,20) = 2 \times 1 + 2 \times 2 + 1 \times 8 = 14$$

أي إلى أن أصل إلى 20 وحدة زمنية سأكون قد تأخرت بمقدار 14 وحدة زمنية.

ثم الآن أحسب من أجل $R(20,34)$

| |
|---|
| 20+14=34 |

$$R(20,34) = input1([20,34[\,) \times C1 + input2([20,34[\,) \times C2 + input3([20,34[\,) \times C3$$
$$= 2 \times 1 + 1 \times 2 + 1 \times 8 = 12$$

وبنفس الطريقة نجد كلاً من :

$$R([34,46[\,) = 3$$
$$R([46,49[\,) = 2$$
$$R([49,51[\,) = 1$$
$$R([51,52[\,) = 0$$

وبالتالي فإن $R4 = 52 < 60 \rightarrow T4$ فالنظام نظام زمن حقيقي.

**ملاحظة:**

في الحالة العامة يجب أن أتأكد من أن جميع المهام لا تتجاوز دورها أي:

$R2 < 12$ و $R3 < 30$ و $R4 < 60$ ليكون النظام نظام زمن حقيقي.

هنا أجرينا الحسابات من أجل $R4$ فقط لطبيعة المثال فقط كما ذكرنا.

إذا وجد عطالة يكون : $R4 + D \leq 60$

$D$ : زمن عطالة، فمثلاً إذا كان يساوي 5 يبقى الشرط محقق حيث $52 + 5 < 60$

**ملاحظة:**

لا نقوم بهذا الحساب (السابق) إلا بعد التأكد من الشروط التالية:

1. تحقق الشرط الثاني، أي $\sum \frac{Ci}{Ti} \leq 1$، وذلك لأنه في حال كانت الحمولة الوسطية أكبر من واحد ستبقى R في تزايد ولن تتناقص أبداً.
2. $R2 < T2 - D2$
3. $R3 < T3 - D3$
4. $R4 < T4 - D4$

لأنه في حال لم يتحقق أحد من هذه الشروط فالنظام ليس نظام زمن حقيقي ولا داعي للقيام بهذا الحساب للتأكد من ذلك.

في حال أردت إضافة العرقلة على الشرط الأكيد السابق :

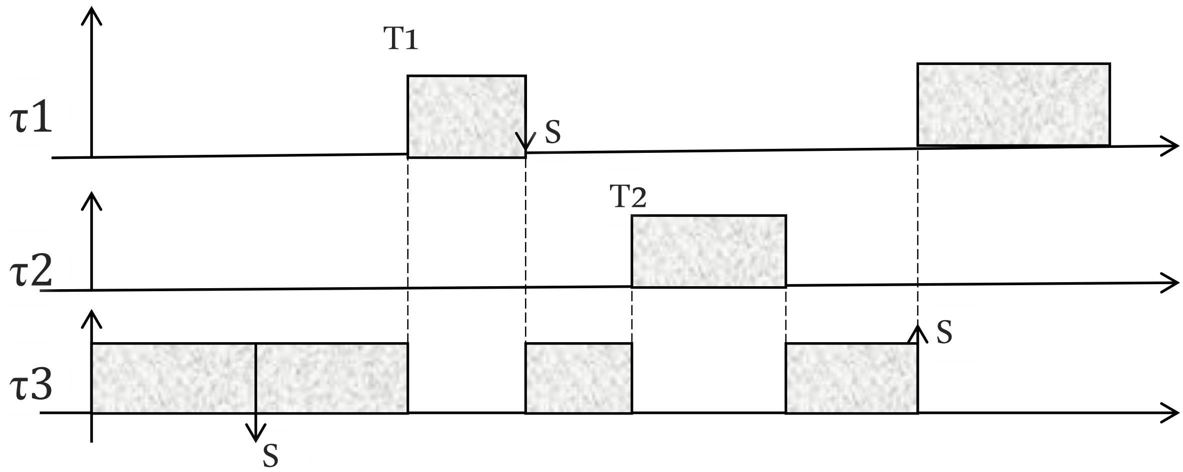
$$R^{n+1}i = Bi + Ci + \sum \left\lceil \frac{Ri^n}{Tj} \right\rceil \times Cj$$

Bi: العرقلة Blocking.

هذا القانون غير فعال إلا بالأنظمة التي تراعي حالة عكس الأولوية ولكن ما معنى عكس الأولوية ؟!

**عكس الأولوية:**

مثال: لدينا المهام $\tau1, \tau2, \tau3$ الأولوية العليا لـ $\tau1$ ثم $\tau2$ ثم $\tau3$، $\tau1$ و $\tau3$ متزامنان على سيمافور واحدة S.




في لحظة من اللحظات كانت $\tau3$ هي التي تنفذ وعندما احتاجت السيمافور أخذتها ثم أتت نبضة $\tau1$ وبما أنها أعلى أولوية من $\tau3$ فستقاطعها وتعمل إلى أن تحتاج لسيمافور S فستطلب السيمافور S ولكن $\tau1$ متزامنة مع $3\tau$ على نفس السيمافور و $\tau3$ كان قد حجز السيمافور *S* فسوف تتوقف $\tau1$ عن التنفيذ ويبقى *Blocking* إلى أن يتم تحرير السيمافور S.

ويعود التنفيذ إلى $\tau3$ ولكن عندما ستأتي نبضة المهمة $\tau2$ وهي أعلى أولوية من $\tau3$ فستقاطعها وتنفذ إلى أن تنتهي تعيد التنفيذ إلى $\tau3$ التي تنهي عملها حتى تحرر السيمافور وبعدها ستنفذ $\tau1$.

إذاً عكس الأولوية هو أن $\tau1$ تنتظر $\tau2$ لتنهي عملها على الرغم من أن $\tau2$ أقل أولوية من $\tau1$ والسبب هو تزامن $\tau1$ مع $\tau3$.

✓ **الحلول:**

الحل الذي تستخدمه لغة الـ *ADA*: أنها تسند أولوية الفعالية إلى أولوية السيمافور التي هي (أولوية السيمافور) ستأخذ أولوية الفعالية الأعلى أولوية ضمن الفعاليات التي عليها.

أي ستصبح أولوية $\tau 3$ من أولوية $\tau 1$.

أي أننا نسند للسيمافور أعلى أولوية فعالية من الفعاليات التي عليها يدوياً، ثم تصبح أي فعالية على هذه السيمافور لها نفس هذه الأولوية.

الحل الذي تستخدمه لغة *java*: يوجد في الجافا *real time task* تعالج هذه المشكلة ديناميكياً تعالجها بما يراه بوراثة الأولوية السيمافور بالجافا هي الـ*monitor* .

يعني إذا كانت $\tau 1$ *blocking* على سيمافور *S* معين كل المهام التي تدخل على هذه السيمافور ترث أولوية هذه المهمة المعرقلة.

$\tau 1$ معرقلة عند سيمافور *S* فإن $\tau 3$ أولويتها من أولوية $\tau 1$.

وهو يماثل الحل السابق ولكن يختلف هنا أنني فقط أرث أولوية المهمة المعرقلة بينما في الـ*ADA* حسب جميع المهام التي على السيمافور.

إذا كان هناك نظام يمنع عكس الأولوية فلا يمكن عرقلة مهمة ذات أولوية أعلى من قبل مهام ذات أولوية أدنى مجتمعة إلا مرة واحدة فقط وإلا يمكن أن تعرقل عدة مرات.

انتهت المحاضرة

**Word press and preparation:**
*Enas Alhalabi*
**Reviewed by:** *Eman Zyadeh*