جامعة المنارة

كلية: .....الهندسة.....

قسم: ..... الهندسة المعلوماتية.....

اسم المقرر: ..... نظم تشغيل 2.....

رقم الجلسة (...4...)

عنوان الجلسة

خوارزميات الجدولة في الزمن الحقيقي

م.عمار مصطفى

الفصل الدراسي العام الدراسي 2024 /2025

جدول المحتويات

Contents

الغاية من الجلسة : التعرف على خوارزميات الجدولة في الزمن الحقيقي (معدل الجدولة الرتيبة وأقرب موعد انتهاء أولا) ، المزايا والعيوب و آلية تنفيذها من خلال مجموعة من المسائل العملية.

خوارزمية معدل الجدولة الرتيبة Rate Monotonic Scheduling (RMS)

ما هي RMS ؟

- RMS هي خوارزمية جدولة (Scheduling Algorithm) تُستخدم في الأنظمة ذات الزمن الحقيقي (RTOS) لتخصيص أولويات للمهام الدورية (Periodic Tasks) وضمان تنفيذها قبل انتهاء مواعيدها النهائية (Deadlines).
- تُعتبر **ثابتة الأولوية** (Static Priority)، حيث تُحدد الأولويات قبل بدء التنفيذ ولا تتغير أثناء التشغيل.
- تُعد **مُسبقة** (Preemptive)، أي أن مهمة ذات أولوية أعلى يمكنها مقاطعة (Preempt) مهمة ذات أولوية أقل إذا أصبحت جاهزة للتنفيذ.

المبادئ الأساسية لـ RMS

1. **المهام الدورية :**
   - كل مهمة (Task) تتكرر بشكل دوري بفترة زمنية ثابتة تُسمى **الفترة** Ti
   - كل مهمة لها **زمن تنفيذ** (Computation Time Ci)، وهو الوقت الذي تحتاجه لإكمال تنفيذها في كل دورة.
   - الموعد النهائي (Deadline) للمهمة هو عادةً نهاية فترتها(Di=Ti) ، أي يجب أن تكتمل المهمة قبل بدء الفترة التالية.
2. **تخصيص الأولويات :**
   - لمهمة ذات **الفترة الأقصر** Ti تُعطى **أولوية أعلى**.
3. **الجدولة المُسبقة :**
   - إذا أصبحت مهمة ذات أولوية أعلى جاهزة، فإنها تُقاطع المهمة الحالية ذات الأولوية الأقل وتأخذ المعالج.
   - عند اكتمال المهمة ذات الأولوية الأعلى، تعود المهمة ذات الأولوية الأقل لاستكمال تنفيذها.
4. **اختبار القابلية للجدولة :(Schedulability Test)**
   - لضمان أن مجموعة المهام يمكن جدولتها بنجاح (أي أن جميع المهام ستُكمل قبل مواعيدها النهائية)، يجب أن تُحقق الاستخدام الكلي للمعالج (CPU Utilization) شرطًا معينًا:

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

حيث :

- n عدد المهام.
- Ci زمن تنفيذ المهمة i
- Ti فترة المهمة i
- U إجمالي استخدام المعالج.

هذا الشرط كافٍ ولكنه ليس ضروريًا. إذا كان U≤1 ، يمكن إجراء تحليل إضافي (مثل تحليل الجدول الزمني) للتأكد من القابلية للجدولة.

الافتراضات :

- المهام مستقلة (لا تتشارك الموارد مثل الأجهزة أو السمافور).
- المواعيد النهائية تتساوى مع الفترات.(Di=Ti)
- لا توجد تكاليف لتبديل السياق (Context Switch) أو تأخير آخر.
- المهام دورية وتبدأ في وقت الصفر (تزامن افتراضي).

كيف تعمل RMS ؟

1. **جمع بيانات المهام :**
   - لكل مهمة، حدد زمن التنفيذ (Ci) والفترة (Ti)
2. **تخصيص الأولويات :**
   - رتب المهام حسب الفترة من الأقصر إلى الأطول، وامنح الأولوية الأعلى للمهمة ذات الفترة الأقصر.
3. **اختبار القابلية للجدولة :**
   - احسب إجمالي استخدام المعالج (U) وقارنه بالحد الأعلى $n(2^{1/n}-1)$

تنفيذ الجدولة :

- عند وقت t=0، ابدأ بتنفيذ المهمة ذات الأولوية الأعلى الجاهزة.
- إذا أصبحت مهمة ذات أولوية أعلى جاهزة، قاطع المهمة الحالية ونفذ المهمة الجديدة.
- استمر في الجدولة مع التحقق من أن كل مهمة تُكمل قبل موعدها النهائي.

مميزات RMS

- **البساطة :**سهلة التنفيذ في الأنظمة ذات الزمن الحقيقي.
- **الأمثلية :**إذا كان هناك أي خوارزمية جدولة ثابتة الأولوية يمكنها جدولة مجموعة مهام، فإن RMS ستتمكن من ذلك أيضًا.
- **الاستقرار :**في حالة الحمل الزائد المؤقت، المهام ذات الأولوية الأعلى ستظل تُنفذ في مواعيدها.

عيوب RMS

- **حد استخدام المعالج :**الحد الأعلى للاستخدام$n(2^{1/n}-1)$ قد يكون منخفضًا (مثل 69% لعدد كبير من المهام)، مما يعني أن بعض الموارد قد لا تُستغل بالكامل.
- انعكاس الأولوية :(Priority Inversion) إذا شاركت المهام الموارد، قد تُعيق مهمة ذات أولوية منخفضة مهمة ذات أولوية عالية، مما يتطلب تقنيات مثل وراثة الأولوية.(Priority Inheritance)
- صعوبة دعم المهام غير الدورية RMS :غير مصممة للمهام العشوائية أو المتقطعة

المسألة 1: مجموعة مهام بسيطة

السؤال :لديك مجموعة من المهام التالية:

Task	Execution time Ci	Time period Ti
T1	1	4
T2	2	6

افترض أن الموعد النهائي يساوي الفترة .(Di=Ti) هل يمكن جدولة هذه المهام باستخدام RMS ؟ إذا كان الجواب نعم، ارسم الجدول الزمني من t=0 إلى t=12.

الحل:

1. **تخصيص الأولويات:**
   - $T_1 = 4 < T_2 = 6$، لذا:
     - $T1$: أولوية عالية.
     - $T2$: أولوية منخفضة.
2. **اختبار القابلية للجدولة:**
   - $n = 2$.
   - $U = \frac{C_1}{T_1} + \frac{C_2}{T_2} = \frac{1}{4} + \frac{2}{6} = 0.25 + 0.333 = 0.583$.
   - الحد الأعلى: $2(2^{\frac{1}{2}} - 1) \approx 0.828$.
   - $0.583 \leq 0.828$، إذن المجموعة قابلة للجدولة.
3. **الجدول الزمني:**

في $t = 0$: $T1, T2$ جاهزتان. $T1$ لها أولوية أعلى، تُنفذ لـ 1 وحدة ($t = 0$ إلى $t = 1$).
في $t = 1$: $T1$ اكتملت، $T2$ تُنفذ لـ 2 وحدة ($t = 1$ إلى $t = 3$).
في $t = 3$: لا مهام جاهزة، المعالج خامل (Idle) حتى $t = 4$.
في $t = 4$: $T1$ جاهزة (فترة جديدة)، تُنفذ لـ 1 وحدة ($t = 4$ إلى $t = 5$).
في $t = 5$: لا مهام جاهزة، خامل حتى $t = 6$.
في $t = 6$: $T2$ جاهزة (فترة جديدة)، تُنفذ لـ 2 وحدة ($t = 6$ إلى $t = 8$).
في $t = 8$: $T1$ جاهزة، تُنفذ لـ 1 وحدة ($t = 8$ إلى $t = 9$).
في $t = 9$: خامل حتى $t = 12$.

Time	0	1	2	3	4	5	6	7	8	9	10	11	12
	T1	T2	T2	Idle	T1	Idle	T2	T2	T1	Idle	Idle	Idle	Idle

- T1: مواعيد نهائية عند t=4,8,12 تكتمل عند t=1,5,9 (جميع المواعيد مُحققة)
- T2: مواعيد نهائية عند t=6,12. تُكتمل عند t=3,8 (جميع المواعيد مُحققة)

الإجابة: المجموعة قابلة للجدولة، والجدول الزمني يُظهر أن جميع المهام تُكتمل قبل مواعيدها النهائية.

المسألة 2: ثلاث مهام مع مقاطعة

السؤال : لديك ثلاث مها مبينة في الجدول التالي:

Task	Execution time Ci	Time period Ti
T1	1	5
T2	2	8
T3	3	10

هل يمكن جدولة هذه المهام باستخدامRMS ؟ إذا كان الجواب نعم، ارسم الجدول الزمني من t=0 إلى t=20

1. **تخصيص الأولويات:**
   - $T_1 = 5 < T_2 = 8 < T_3 = 10$.
   - الأولويات: $T1$ (أعلى)، $T2$، $T3$ (أدنى).
2. **اختبار القابلية للجدولة:**
   - $n = 3$.
   - $U = \frac{C_1}{T_1} + \frac{C_2}{T_2} + \frac{C_3}{T_3} = \frac{1}{5} + \frac{2}{8} + \frac{3}{10} = 0.2 + 0.25 + 0.3 = 0.75$.
   - الحد الأعلى: $3(2^{\frac{1}{3}} - 1) \approx 3(1.26 - 1) \approx 0.78$.
   - $0.75 \leq 0.78$، إذن المجموعة قابلة للجدولة.

- في t=0: T1,T2,T3 جاهزة. T1 (أولوية أعلى) تُنفذ لـ 1 وحدة t=0 إلى t=1
- في t=1: T2 تُنفذ لـ 2 وحدة t=1 إلى t=3
- في t=3: T3 تُنفذ لـ 3 وحدة t=3 إلى t=6
- في t=5 : T1 جاهزة، تُقاطع T3، تُنفذ لـ 1 وحدة t=5 إلى t=6
- في t = 6: T3 تستأنف (كانت مُقاطعة)، لكن T2جاهزة (أولوية أعلى)، فتُنفذ T2 لـ 2 وحدة t=6 إلى t=8
- في t=8: T3 تستأنف، تُنفذ لـ 2 وحدة متبقية t=8 إلى t=10
- في t=10: T1 جاهزة، تُنفذ لـ 1 وحدة t=10 إلى t=11
- في t=11: خامل حتى t=13
- في t=13: T2 جاهزة، تُنفذ لـ 2 وحدة t=13 إلى t=15
- في t=15: T1 جاهزة، تُنفذ لـ 1 وحدة t=15 إلى t=16
- في t=16: T3 جاهزة، تُنفذ لـ 3 وحدة t=16 إلى t=19
- في t=19: خامل حتى t=20

Time	0	1	2	3	4	5	6	7	8	9	10	11	12
	T1	T2	T2	T3	T3	T1	T3	idle	T2	T2	T1	T3	T3

13	14	15	16	17	18	19	20
T3	idle	T1	T2	T2	idle	Idle	T1

التحقق من المواعيد:

- T1مواعيد عند t = 5, 10, 15, 20 تُكتمل عند t = 1, 6, 11, 16 مُحققة.
- T2مواعيد عند t = 8, 16 تُكتمل عند t = 3 , 8, 15 مُحققة.
- T3مواعيد عند t = 10, 20 تُكتمل عند t = 10, 19 محققة.

الإجابة: المجموعة قابلة للجدولة، والجدول الزمني يُظهر تنفيذ جميع المهام قبل مواعيدها.

المسألة 3: مجموعة مهام غير قابلة للجدولة

السؤال :لديك مهمتين:

Task	Execution time Ci	Time period Ti
T1	25	50
T2	35	80

هل يمكن جدولة هذه المهام باستخدامRMS ؟ إذا كان الجواب نعم، ارسم الجدول الزمني اذا كان لا وضح السبب.

1. **تخصيص الأولويات:**
   - $T_1 = 50 < T_2 = 80.$
   - الأولويات: $T1$ (أعلى)، $T2$ (أدنى).
2. **اختبار القابلية للجدولة:**
   - $n = 2.$
   - $U = \frac{C_1}{T_1} + \frac{C_2}{T_2} = \frac{25}{50} + \frac{35}{80} = 0.5 + 0.4375 = 0.9375.$
   - الحد الأعلى: $2(2^{\frac{1}{2}} - 1) \approx 0.828.$
   - $0.9375 > 0.828$، لذا الشرط الكافي لم يتحقق. لكن يجب إجراء تحليل الجدول الزمني لأن $U < 1.$
3. **الجدول الزمني (للتحقق):**

في $t = 0$: $T1, T2$ جاهزتان. $T1$ تُنفذ لـ 25 وحدة ($t = 0$ إلى $t = 25$).
في $t = 25$: $T2$ تُنفذ لـ 25 وحدة ($t = 25$ إلى $t = 50$).
في $t = 50$: $T1$ جاهزة (فترة جديدة)، $T2$ تُقاطع، تُنفذ لـ 25 وحدة ($t = 50$ إلى $t = 75$).
في $t = 75$: $T2$ تستأنف، تحتاج 10 وحدات متبقية، تُنفذ من $t = 75$ إلى $t = 85$.
المشكلة: موعد $T2$ النهائي عند $t = 80$، لكن $T2$ تُكتمل عند $t = 85$. هذا يعني أن $T2$ فشلت في تحقيق موعدها النهائي.

Time	0	25	50	75	80	85	100
	T1(25)	T2(25)	T1(25)	T2(10)	Idle		

Deadline T2 (missed at t=80)

التحقق من المواعيد:

- T1: مواعيد عند t=50,100 تُكتمل عند t=25,75 مُحققة
- T2: موعد عند t=80 تُكتمل عند t=85 غير مُحققة

الإجابة :المجموعة **غير قابلة للجدولة** باستخدام RMS لأن T2 تفشل في تحقيق موعدها النهائي عند t=80. السبب هو أن استخدام المعالج (0.9375) يتجاوز الحد الأعلى (0.828)، وتحليل الجدول الزمني يُظهر أن T1 ذات الأولوية العالية تُقاطع T2، مما يؤخر اكتمالها

السؤال : لديك ثلاث مها مبينة في الجدول التالي:

Task	Arrival Time	Execution time Ci	Time period Ti
T1	0	1	5
T2	3	2	8
T3	5	3	10

هل يمكن جدولة هذه المهام باستخدامRMS ؟ إذا كان الجواب نعم، ارسم الجدول الزمني من t=0 إلى t=20

الخطوة 1: فهم جدولة التواتر الثابت (RMS) وخصائص المهام

جدولة التواتر الثابت :(RMS) هو خوارزمية جدولة ذات أولوية ثابتة، حيث تُمنح المهام ذات الفترات الأقصر (التردد الأعلى) أولوية أعلى. في هذه الحالة:

- تتمتع المهمة T1 بالفترة الأقصر(Ti = 5) ، وبالتالي لها الأولوية الأعلى.
- تتمتع المهمة T2 بفترة 8، وبالتالي لها الأولوية الثانية من حيث الارتفاع.
- تتمتع المهمة T3 بالفترة الأطول(Ti = 10) ، وبالتالي لها الأولوية الأدنى.

خصائص المهام:

- **وقت الوصول :(Arrival Time)** اللحظة التي تصل فيها المهمة وتصبح جاهزة للتنفيذ.
- **وقت التنفيذ :(Ci)** المدة التي تحتاجها المهمة لإكمال تنفيذها.
- **فترة الوقت :(Ti)** الفاصل الزمني الذي تتكرر فيه المهمة (المصدر الزمني لكل حالة).

يجب أن تكتمل كل حالة من المهام قبل المصدر الزمني الخاص بها) الذي يساوي الفترة بالنسبة للحالة الأولى، مع مصادر زمنية لاحقة عند فواصل تساوي.(Ti

---

الخطوة 2: التحقق من قابلية الجدولة باستخدام RMS

لتحديد ما إذا كانت المهام يمكن جدولتها، نحسب أولاً **معدل الاستخدام (Utilization)** ونقارنه بحد جدولة.RMS

حساب معدل الاستخدام

يُعطى معدل الاستخدام U لمهمة بواسطة Ci/Ti ويكون المجموع الكلي لمعدل الاستخدام هو مجموع معدلات الاستخدام الفردية:

- T1 لـ $U_1 = \frac{1}{5} = 0.2$
- T2 لـ $U_2 = \frac{2}{8} = 0.25$
- T3 لـ $U_3 = \frac{3}{10} = 0.3$

الاستخدام الكلي:

$$U = U_1 + U_2 + U_3 = 0.2 + 0.25 + 0.3 = 0.75$$

حد جدولة RMS

بالنسبة لـ n مهام، يُعطى حد جدولة RMS بواسطة:

$$U_{\text{bound}} = n\left(2^{\frac{1}{n}} - 1\right)$$

هنا، $n = 3$:

$$U_{\text{bound}} = 3\left(2^{\frac{1}{3}} - 1\right)$$

$$2^{\frac{1}{3}} \approx 1.2599 \quad \text{ومن ثم} \quad 2^{\frac{1}{3}} - 1 \approx 0.2599$$

$$U_{\text{bound}} = 3 \times 0.2599 \approx 0.7797$$

بما أن الاستخدام الكلي U = 0.75 أقل من الحد 0.7797، فإن المهام قابلة للجدولة نظريًا تحت RMS. ومع ذلك، يجب أيضًا مراعاة أوقات الوصول وإجراء تحليل زمني لتأكيد الجدوى على المجال من t = 0 إلى t = 20.

الخطوة 3: تحليل الجدول الزمني (من t=0 الى t=20):

سنقوم بمحاكاة الجدولة بمراعاة وصول المهام، فتراتها، وأولوياتها. ستولد كل مهمة حالات عند فواصل تساوي فترتها، وسنقوم بجدولتها بناءً على أولويات RMS (T1 > T2 > T3).

حالات المهام

- **T1** (الفترة = 5، Ci = 1): الحالات تصل عند $t = 0, 5, 10, 15, 20$.
- **T2** (الفترة = 8، Ci = 2): الحالات تصل عند $t = 3, 11, 19$.
- **T3** (الفترة = 10، Ci = 3): الحالات تصل عند $t = 5, 15$.

الجدول الزمني

- **t=0**: تصل T1 (T1.1). تُنفذ لمدة وحدة واحدة (من t=0 إلى t=1). لا توجد مهام أخرى.
- **t=1 إلى t=3**: خامل (لا توجد مهام جاهزة).
- **t=3**: تصل T2 (T2.1). تُنفذ لمدة وحدتين (من t=3 إلى t=5).
- **t=5**: تصل T1 (T1.2) وT3 (T3.1).
  - T1 لها الأولوية الأعلى، تُنفذ لمدة وحدة واحدة (من t=5 إلى t=6).
- **t=6**: T2 تم قطع (تم إكمالها بالفعل)، تُنفذ T3 (T3.1) لمدة 3 وحدات (من t=6 إلى t=9).
- **t=9 إلى t=10**: خامل.
- **t=10**: تصل T1 (T1.3)، تُنفذ لمدة وحدة واحدة (من t=10 إلى t=11).
- **t=11**: تصل T2 (T2.2)، تُنفذ لمدة وحدتين (من t=11 إلى t=13).
- **t=13 إلى t=15**: خامل.
- **t=15**: تصل T1 (T1.4) وT3 (T3.2).

تُنفذ T1 لمدة وحدة واحدة (من t=15 إلى t=16).

- **t=16**: تُنفذ T3 (T3.2) من t=16 إلى t=19).
- **t=19**: تصل T2 (T2.3)، تُنفذ لمدة وحدتين من t=19 إلى t=21). ملاحظة: توقف التحليل عند t=20.
- **t=20**: تصل T1 (T1.5)، لكننا توقف التحليل هنا.

التحقق من المصادر الزمنية

- **T1**: المصادر الزمنية عند t=5, 10, 15, 20. تكتمل جميع الحالات (T1.1 إلى T1.4) في الوقت المحدد t=1, 6, 11, 16).
- **T2**: المصادر الزمنية عند t=11 (T2.1)، t=19 (T2.2)، t=27 (T2.3). تكتمل T2.1 عند t=5، T2.2 عند t=13، وكلاهما في الوقت المحدد. تبدأ T2.3 عند t=19، تحتاج إلى وحدتين، لكن وحدة واحدة فقط متاحة عند t=20) تفوت المصدر الزمني عند t=27، ولكننا نتحقق فقط حتى t=20).
- **T3**: المصادر الزمنية عند t=15 (T3.1)، t=25 (T3.2). تكتمل T3.1 عند t=9، T3.2 عند t=19، وكلاهما في الوقت المحدد.

الخطوة 4: الاستنتاج

- حتى $t = 20$، تلتقي جميع المهام بمصادرها الزمنية ضمن الفاصل المنقح. تبدأ T2.3 عند t=19 وتحتاج إلى وحدتين، لذا تكتمل عند t=21، ولكن بما أن المصدر الزمني الخاص بها عند t=27 ونحن نتحقق فقط حتى t=20، نعتبر الجدولة قابلة للتطبيق للفاصل المعطى.
- اختبار الاستخدام يؤكد القابلية النظرية للجدولة ($U = 0.75 > 0.7797$)، وتحليل الجدول الزمني لا يظهر تفويتًا لمصادر زمنية ضمن $t = 0$ إلى $t = 20$.

الإجابة: نعم، يمكن جدولة المهام باستخدام RMS من $t = 0$ إلى $t = 20$.

خوارزمية Earliest Deadline First (EDF) Scheduling في أنظمة الزمن الحقيقي

خوارزمية **EDF** هي إحدى خوارزميات الجدولة المستخدمة في أنظمة الزمن الحقيقي (Real-Time Systems) لتخصيص الموارد (مثل وحدة المعالجة المركزية) للعمليات أو المهام بناءً على **المواعيد النهائية** (Deadlines). تُعتبر هذه الخوارزمية **ديناميكية** و**مثلى** في ظل ظروف معينة، مما يعني أنها تحدد الأولويات بشكل مستمر بناءً على الموعد النهائي الأقرب

مبادئ عمل الخوارزمية

1. **الأولوية بناءً على الموعد النهائي** :
   - كل مهمة (Task) لها **موعد نهائي** (Deadline)، وهو الوقت الذي يجب أن تكتمل فيه المهمة.
   - المهمة ذات **الموعد النهائي الأقرب** تحصل على الأولوية العليا ويتم تنفيذها أولاً.
2. **الجدولة الديناميكية** :
   - الأولويات ليست ثابتة، بل تتغير بمرور الوقت بناءً على المواعيد النهائية النسبية (Relative Deadlines) للمهام.
   - في كل لحظة زمنية، يتم اختيار المهمة التي لها أقرب موعد نهائي.

الافتراضات الأساسية :

- المهام دورية (Periodic) أو غير دورية.(Aperiodic)
- كل مهمة لها **مدة تنفيذ** (Execution Time)و**موعد نهائي** (Deadline)و**دورة** (Period)إذا كانت دورية.
- النظام يدعم **الانقطاع المسبق** (Preemption)، أي يمكن إيقاف مهمة لتنفيذ مهمة أخرى ذات أولوية أعلى.

الأمثلية :

- EDFهي خوارزمية مثالية لجدولة المهام على معالج واحد إذا كان إجمالي استخدام المعالج (Utilization) لا يتجاوز 100.(U ≤ 1) %
- أي أنها قادرة على جدولة جميع المهام دون تفويت أي موعد نهائي إذا كانت الشروط مناسبة

خطوات عمل الخوارزمية

1. **جمع بيانات المهام** :
   - لكل مهمة :
     - Ci زمن التنفيذ.(Execution Time)
     - Di الموعد النهائي النسبي.(Relative Deadline)
     - Ti الدورة (Period) إذا كانت المهمة دورية.
   - على سبيل المثال: مهمة T1 لها C1=2 ، D1=5، T1=5
2. حساب استخدام المعالج:(Utilization)

- لكل مهمة: $U_i = \frac{C}{T}$.
- إجمالي الاستخدام: $U = \sum U_i$.
- إذا كان $U \geq 1$، فإن المهام قابلة للجدولة باستخدام EDF.

التحقق من الجدولة :

- تأكد أن كل مهمة تكتمل قبل موعدها النهائي.
- إذا تم تفويت أي موعد نهائي، فإن الجدولة غير ناجحة.

المسألة:

لنفترض أن لدينا ثلاث مهام دورية:

Task	Execution Time Ci	Periord Time Ti	Deadline
T1	1	4	4
T2	2	6	6
T3	3	12	12

حساب الاستخدام

- $U_1 = \frac{C_1}{T_1} = \frac{1}{4} = 0.25.$
- $U_2 = \frac{C_2}{T_2} = \frac{2}{6} = 0.333.$
- $U_3 = \frac{C_3}{T_3} = \frac{3}{12} = 0.25.$
- $U = U_1 + U_2 + U_3 = 0.25 + 0.333 + 0.25 = 0.833.$
- بما أن $1U \leq$، فالمهام قابلة للجدولة.

الخطوة 2: الجدولة من t=0 إلى t=12

- عند t=0
  - المهام الجاهزة(D1=4) T1 :، (D2=6) T2، .(D3=12) T3
  - أقرب موعد نهائي.(D1=4) T1 :
  - نفّذ T1 من t=0 إلى t=1
- عند t=1
  - المهام الجاهزة(D2=6) T2 :، .(D3=12) T3
  - أقرب موعد نهائي.(D2=6) T2 :
  - نفّذ T2 من t=1 إلى t=3
- عند t=3
  - المهام الجاهزة.(D3=12) T3 :
  - نفّذ T3 من t=3 إلى t=6
- عند t=4
  - تصل نسخة جديدة من.(D1=8) T1
  - أقرب موعد نهائي.(D1=8) T1 :
  - قم بالانقطاع ونفّذ T1 من t=4 إلى t=5
- وهكذا تستمر الجدولة مع مقارنة المواعيد النهائية في كل خطوة.

النتيجة:

- يتم إنشاء جدول زمني يضمن اكتمال كل مهمة قبل موعدها النهائي.

Time	0	1	2	3	4	5	6	7	8	9	10	11	12
	T1	T2	T2	T3	T1	T3	T2	T2	T1	T2	T2	T3	T1

المسألة:

لنفترض أن لدينا ثلاث مهام دورية:

Task	Execution Time Ci	Periord Time Ti	Deadline
T1	3	5	5
T2	2	4	4

- **الفاصل الزمني** :من t=0 الى t=20

الخطوة 1: حساب معدل الاستخدام

- T1: $\frac{3}{5} = 0.6$
- T2: $\frac{2}{4} = 0.5$

$$U = 0.6 + 0.5 = 1.1$$

$U > 1$، لذا المهام غير قابلة للجدولة نظريًا باستخدام EDF.

الخطوة 2: محاولة الجدولة لتوضيح الفشل

- **T1** (Ti = 5): المواعيد النهائية: 5, 10, 15. تصل عند $t = 0, 5, 10$.
- **T2** (Ti = 4): المواعيد النهائية: 4, 8, 12, 16. تصل عند $t = 0, 4, 8, 12$.

الخطوة 3: الجدولة الزمنية

- **t=0**: T1 (T1.1) وT2 (T2.1) تصلان.
  - موعد نهائي T1.1 = 5، T2.1 = 4.
  - T2.1 لها الموعد النهائي الأقرب، تُنفذ من $t = 0$ إلى $t = 2$.
- **t=2**: T1.1 جاهزة ($t = 5$). تُنفذ من $2t =$ (الموعد النهائي 5) إلى T1.1.
- **t=5**: T1 (T1.2) وT2 (T2.2) تصلان.
  - موعد نهائي T2.2 = 8، T1.2 = 10.
  - T2.2 لها الموعد النهائي الأقرب، لكنها تحتاج إلى وحدتين، تُنفذ من $t = 5$ إلى $t = 7$.
- **t=7**: T1.2 جاهزة ($t = 10$). تُنفذ من $7t =$ (الموعد النهائي 10) إلى T1.2.
- **t=8**: T2 (T2.3) تصلت عند $8t =$، الموعد النهائي 12، لكن T1.2 لا تزال تُنفذ.
- **t=10**: T2.3 جاهزة، لكنها تفوت الموعد النهائي لأنها تحتاج إلى وحدتين ولم تُنفذ بعد.

الخطوة 4: التحقق من المواعيد النهائية

- **T1**: تُكمل عند $10t = 5,$ في الوقت.
- **T2**: لم تُنفذ بحلول $12t =$، تفوت الموعد T2.3 عند $7t =$، لكن T2.2 عند $t = 2$، المواعيد النهائية T2.1 عند 4, 8, 12. تُكمل النهائي.

النتيجة:

- المهام غير قابلة للجدولة.
- **التحليل**: معدل الاستخدام $U = 1.1 > 1$، مما يعني أن المعالج لا يستطيع التعامل مع الحمل. هذا يؤدي إلى تراكم المهام وتفويت المواعيد النهائية.