نمذجة استراتيجية تخزين نقاط الاستعادة/الاسترجاع للوصول إلى أفضل زمن تنفيذ في التطبيقات المتوازية

د.سمير جعفر*

د.محمد مناف الحمد

رهف غزال***

(تاريخ الإيداع 13 / 1 / 2019. قُبِل للنشر في 26 / 6 /2019)

□ ملخّص □

نقدّم في هذا البحث نموذج رياضي لآلية تخزين نقاط الاستعادة / الاسترجاع (checkpoint/recovery)، بهدف ضمان انتهاء تنفيذ التطبيق المتوازي في منصات العمل الحسابية ذات الأداء العالي **وبأقل زمن ممكن**، والذي يتحقق من خلال تقليل كلفة الخسارة الحسابية نتيجة الأعطال المتوقعة التي قد تصيب مكونات النظام من جهة، والحدّ من الكلفة الزائدة لآليات التسامح مع الأعطال خلال العمل من جهة أخرى.

نهتم في بحثنا بنوع خاص من الأعطال قد يصيب المكونات يسمى عطل التوقّف (crash fault) والذي يبدي سلوك ثابت للنظام أثناء التنفيذ إما الفشل أو العمل في لحظة ما، ولتحقيق استمرارية تنفيذ التطبيق بالرغم من حدوث الأعطال ندرس استراتيجية نقاط الاستعادة المتناسقة كآلية للتسامح مع الأعطال.

الكلمات المفتاحية: نقاط الاستعادة المتناسقة، التسامح مع الأعطال، عطل التوقف، نموذج رياضي.

***أستاذ مساعد – قسم الرياضيات– كلية العلوم – جامعة دمشق – دمشق – سورية.**

****** أستاذ – قسم الرياضيات– كلية العلوم – جامعة دمشق – دمشق – سورية.**

******* طالبة دراسات عليا (دكتوراه)– قسم الرياضيات – كلية العلوم – جامعة دمشق – دمشق – سورية.**

Modeling of checkpointing/rollback strategy towards optimal run time in parallel applications

Dr. Samir Jafar*
Dr. Mohammed Mounaf Al-hamad**
Rahaf Ghazal***

(Received 13 / 1 / 2019. Accepted 26 / 6 /2019)

□ ABSTRACT □

We present a mathematical model of checkpointing/rollback strategy, in order to ensure that execution of parallel applications in High Performance Computing (HPC) platform are completed in as little time as possible, which is achieved through minimize the computations loss due to expected failures or unnecessary overhead of fault tolerant mechanisms.

In our study, we are interested in special failure of components, which is called (crash fault), that shows a constant behavior of system during the work, either failure or work at for a moment, and we study a coordinated checkpointing strategy for fault tolerance to achieve continuity of the application despite the failures.

key words: coordinated checkpointing, fault tolerance, crash fault, mathematical model.

*** Associate Professor, at Department of Mathematics, Faculty of Science, Damascus University, Damascus, Syria.**

**** Professor, at Department of Mathematics, Faculty of Science, Damascus University, Damascus, Syria.**

***** Phd's Student, Department of Mathematics, Faculty of Science, Damascus University, Damascus, Syria.**

مقدمة:

يُستخدم نظام الحوسبة الشبكية grid computing system كمنصة عمل حسابية ضخمة ومهمّة لحل مسائل الحسابات الطويلة والمعقدة [1]، بالمقابل فإنّ اتساع هذه النظم جعلها عُرضةً للتحديات المتعلّقة بتحقيق موثوقية تطبيقاتها (reliability) وتوفّر مكوناتها (availability) خلال العمل [2]، فكان لابدّ من تزويد التطبيقات بآليات للتسامح مع الأعطال لتحقيق استمرارية تنفيذها في ظل وجود أعطال مكونات النظام؛ من أشهر هذه الآليات استرجاع حالة سابقة للنظام ــ تم تخزينها خلال التنفيذ الخالي من الأخطاء ــ بعد حدوث الفشل، حيث يتم تخزين هذه الحالات بشكل متكرر ضمن ذاكرة مستقرة وتسمّى هذه العملية بحفظ نقاط الاستعادة (checkpointing) [3].

بالمقابل كان لابد من دراسة تحليلية دقيقة للكلفة الزمنية المرافقة لتلك الآليات كي لا تكون حِمْلاً جديداً يضاف إلى زمن تنفيذ التطبيق في البيئة [4]، وبالتالي نسعى في دراستنا إلى تحديد أفضل وقت لتخزين نقاط الاستعادة بغية الحصول على أقل زمن تنفيذ للتطبيق المتوازي في بيئة الحوسبة الشبكية؛ في ضوء الدراسات المُقدَّمة في هذا المجال[5] وجدنا أنّ نمذجة استراتيجية تسجيل نقاط الاستعادة للتطبيقات المتوازية تتم دراستها إمّا بالاعتماد على نموذج دالة الكلفة[5]، بحيث يكون الهدف الحصول على نموذج رياضي يحدد كلفة الخسارة الزمنية المتوقعة(كلفة تخزين نقاط الاستعادة واستعادة التنفيذ) ومن ثم محاولة تصغير هذه الكلفة، أو على نماذج الموثوقية[5] والتي تهدف إلى نمذجة استراتيجية تسجيل نقاط الاستعادة للحصول على أعلى موثوقية ممكنة للتطبيق.

سنهتم في دراستنا بنمذجة دالة الكلفة المتعلقة بزمن تنفيذ التطبيق والتي نحصل عليها من خلال نمذجة العطل المُستهدَف(crash fault) في نظام الحوسبة الشبكية، ومن ثم الوصول إلى علاقة رياضية تحدد القيمة المتوقعة لزمن انتهاء تنفيذ التطبيق بوجود تلك الأعطال وذلك باستخدام آلية للتسامح مع الأعطال تعتمد على تخزين نقاط الاستعادة المتناسقة للتطبيق (coordinated checkpointing)، وأخيراً قمنا بحلّ هذا النموذج الرياضي لتحديد السلسلة الزمنية الأمثل لتسجيل نقاط الاستعادة المتناسقة ومناقشة النتائج الرياضية التي حصلنا عليها ومطابقتها مع أرض الواقع.

أهمية البحث وأهدافه:

نهدف للوصول إلى أفضل زمن تنفيذ ممكن للتطبيق في بيئة الحوسبة الشبكية على الرغم من تعطّل مكوناتها أثناء العمل، وذلك من خلال نموذج رياضي يصف تنفيذ التطبيق المتوازي وآلية التسامح مع الأعطال(fault tolerance mechanism) المستخدمة، وبذلك يتم تقديم أنسب أداء لآلية تخزين نقاط الاستعادة وفقاً للمعايير المطلوبة من حيث موثوقية التطبيق وجهوزية مكونات البيئة.

1) تحليل الموثوقية في النظم ذات الأداء العالي: reliability analysis in HPS:

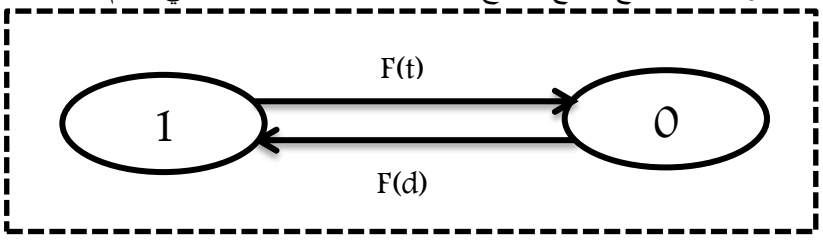
يتضمن هذا القسم تحديد شكل الخطأ الذي تتعرض له مكونات بيئة الحوسبة الشبكية وطريقة تكيّفها معه، ومن ثم نمذجة العطل باستخدام سلاسل ماركوف للوصول إلى علاقة رياضية تحدد التوقع الرياضي لزمن انتهاء تنفيذ التطبيق باستخدام آلية نقاط استعادة متناسقة.

1.3. نمذجة العطل:

تتعرض مكونات بيئة الحوسبة الشبكية إلى أشكال مختلفة من الأخطاء تؤدي إلى خلل في أدائها[6]، فمنها ما يكون سبباً لتوقف تنفيذ التطبيق ومنها ما يكون سبباً للحصول على نتائج خاطئة؛ سنهتم في دراستنا بالأخطاء المادية الدائمة

(crash fault) والتي تؤدي إلى توقف التنفيذ في حال حدوثها، وبالتالي يمكن وصف النظام وفق حالتين فقط إما أنه يعمل وسنرمز لهذه الحالة بالحالة 1 أو أنه قد توقف عن التنفيذ ونعبر عن ذلك بالحالة 0.

إذاً وباستخدام سلاسل ماركوف نستطيع وضع نموذج لحالة وحدة الحساب الواحدة في نظام الحوسبة الشبكية كما يلي :



الشكل 1: نموذج ماركوف لوحدة الحساب الواحدة

نقوم باستخدام سلاسل ماركوف[7] ذو فضاء حالات منتهي والانتقالات الزمنية بين الحالات تأخذ أي قيمة حقيقة، نرمز لفضاء الحالة بالرمز S وهو على الشكل : $S = \{0 , 1\}$.

من الشكل (1) نلاحظ أنه قد تم التعبير عن أزمنة الانتقال بين الحالتين بمتغيرات عشوائية مستمرة: T الزمن حتى حدوث العطل و D الزمن حتى عودة المكون إلى العمل، وسنفرض بأنّ كليهما متكافئان من حيث نوع التوزيع الاحتمالي الذي يتبعان له، وبالتالي فإن F(t) و F(d) تمثلان دوال التوزيع الاحتمالي للمتغيرين T, D على الترتيب.

لاختيار التوزيع الاحتمالي المناسب لهذين المتغيرين العشوائيين علينا تحديد سلوك المكون في حالة الفشل بشكل دقيق وكيفية تعامل النظام مع هذا الحدث من خلال القواعد التالية:

- يمر النظام بحالتين أساسيتين خلال التنفيذ هما العمل والفشل.
- نعرّف الفشل بأنه توقف النظام عن تنفيذ التطبيق الحالي ضمنه والذي يستدعي عملية الاسترجاع (recovery).
- فشل مكون واحد على الأقل في النظام يؤدي إلى فشل التنفيذ الحالي للنظام.
- عمل جميع مكونات النظام معاً في لحظة معينة يعني أن النظام يعمل في تلك اللحظة.
- بعد الفشل فإنّ النظام قابل لإعادة التشكيل (reconfiguration) إمّا بالعدد نفسه للمكونات أو بعدد مختلف، دون أيّة شروط على عدد المكونات بعد التشكيل الجديد.
- بعد الفشل يسترجع النظام حالته من حالة تنفيذ خالية من الأخطاء قد تم تخزينها سابقاً تُدعى بنقطة الاستعادة (checkpoint).
- نقاط الاستعادة المخزنة من نوع نقاط الاستعادة المتناسقة coordinated checkpoints [8]والتي تعتمد على حفظ متكرر للتنفيذ (عادة يتم بشكل دوري) في نقطة استعادة واحدة تشمل معلومات عن الحالة العامة المتناسقة للتطبيق كاملاً وضمن ذاكرة مستقرة.
- تسجيل نقطة الاستعادة هو جزء من مهام المعالجات المشاركة حيث تتم خلال التنفيذ بأمر من المنسّق (coordinator) وبالتالي قد يحصل الخطأ خلال التسجيل وتفشل مهمة التخزين وقد تتم بنجاح إذا لم يطرأ حدث ما ضمن النظام.
- بعد الفشل يتابع النظام تنفيذ التطبيق على مكوناته وكأنها جديدة.

وفقاً للشرط الأخير من الشروط السابقة فإن أفضل توزيع احتمالي يمكن أن يمثل فشل النظام هو التوزيع الأسي[9]، ذلك أنّ زمن توقف المكون الذي يتبع لتوزيع أسي لا يتأثر بكمية الزمن الذي قضاه المكون في العمل سابقاً قبل التوقف، فإذا توقف في لحظة ما يُكمل عمله مرة أخرى بعد الإصلاح وكأنّه جديد، ونقول في هذه الحالة أن التوزيع الاحتمالي لا يملك ذاكرة (Stochastic Renewal Process)؛ في العديد من التطبيقات يتم اختيار التوزيع الأسي كتوزيع لزمن التوقف كونه بسيط رياضياً من جهة ويمثل العمل الواقعي بشكل كافي من جهة أخرى.

إذاً يحدث الانتقال بين حالات ماركوف وفق توزيع أسي وبمعدلات ثابتة؛ μ معدل الإصلاح (معدل الانتقال من 0 إلى 1)و λ معدل الفشل (معدل الانتقال من 1 إلى 0).

2.3. النموذج الرياضي لزمن انتهاء التنفيذ بدون آلية للتسامح مع الأعطال:

في البيئات التفرّعية الحسابية عالية الأداء مثل بيئة الحوسبة الشبكية والتي تستخدم آليات للتسامح مع الأعطال، فإنّ زمن تنفيذ التطبيق يتأثر حكماً بالزمن الذي يقضيه النظام في العمل المفيد حتى حدوث عطل ما ومن ثم استعادة التنفيذ بعد الإصلاح، بالإضافة لزمن تخزين نقاط الاستعادة خلال التنفيذ الخالي من الأخطاء، مما يزيد من الوقت المتوقع لانتهاء تنفيذ التطبيق.

بدايةً النموذج الرياضي لزمن التنفيذ بدون استخدام آلية للتسامح مع الأعطال يعطى بالشكل التالي[13]:

$$T(w) = \sum_{i=1}^{nf_w} \left[\Delta t_i^{(f)} + \Delta t_i^{(R)}\right] + w \qquad (1)$$

تعّبر العلاقة السابقة عن زمن تنفيذ التطبيق المتوازي المكون من w وحدة عمل بدون آلية للتسامح مع الأعطال حيث نلاحظ أنه قد أُضيف للزمن الأصلي مجموع عدد مرات إعادة التنفيذ نتيجة الأخطاء الحاصلة خلال التنفيذ.

حيث :

nf_w هو عدد الأعطال الحاصلة حتى انتهاء تنفيذ التطبيق المكون من w وحدة عمل.

$\Delta t_i^{(f)}$ هي متوسط الفترة الزمنية حتى تعطّل المكون للمرة رقم i .

$\Delta t_i^{(R)}$ هي متوسط الفترة الزمنية حتى إصلاح المكوّن للمرة رقم i.

لتقدير الفترات الزمنية السابقة نفرض ما يلي:

لدينا مكون مُمَثَّل باستخدام سلاسل ماركوف بزمن مستمر وفق فضاء حالة منتهي S={1,…,r}، فإنَّ احتمال انتقال هذا المكون إلى الحالة j مع العلم بأنه في اللحظة الابتدائية (t=0) كان في الحالة i يُعطى بالعلاقة:

$$p_{ij}(t) = pr(T(t) = j \mid T(0) = i) \;\; ; i,j \in S \;\; , n(s) = r$$

حيث يشير $n(s)$ إلى عدد عناصر فضاء العينة S، وT متغير عشوائي يدل على زمن الانتقال .

بما أننا سوف نستخدم التوزيع الأسي لتمثيل زمن الانتقال بين حالات ماركوف فإنّ الاحتمال السابق يمكن لنا أن نكتبه بالشكل:

$$p_{ij}(\Delta t) = pr\left(T_{ij} \le \Delta t\right) = 1 - e^{-a_{ij}\Delta t} \cong 1 - \left(1 - a_{ij}\Delta t\right) = a_{ij}\Delta t \qquad (2)$$
$$; \; \Delta t = t_j - t_i$$

علماً أنّ a_{ij} يمثل معدل الانتقال من الحالة i إلى الحالة j؛ تجدر الإشارة إلى أن التقريب السابق تم الحصول عليه من منشور ماك لوران للتابع الأسي وذلك بالاكتفاء بأول حدّين فقط.

بالاستفادة من العلاقة (2) نستطيع أن نكتب :

$$\Delta t \cong \frac{p_{ij}(\Delta t)}{a_{ij}}$$

إذاً فعند حدوث خطأ ما في الفترة Δt فإن: $p_{10}(\Delta) \approx 1 \quad ; \forall \Delta t$

$$=> \Delta t_k^{(f)} \cong \frac{p_{10}\left(\Delta t^{(f)}\right)}{a_{10}} = \frac{1}{a_{10}} = \frac{1}{\lambda_k}$$
$$= MTTF_k(mean\ time\ to\ failuer\ for\ the\ k^{th} component)$$

وعند إصلاح المكون وعودته إلى العمل في الفترة Δt أيضاً يكون: $: p_{01}(\Delta t) \approx 1 \quad ; \forall \Delta t$

$$=> \Delta t_k^{(R)} \cong \frac{p_{01}\left(\Delta t^{(R)}\right)}{a_{01}} = \frac{1}{a_{01}} = \frac{1}{\mu_k}$$
$$= MTTR_k(mean\ time\ to\ repair\ for\ the\ k^{th} component)$$

إذاً تصبح العلاقة (1) بالشكل:

$$T(w) = \left[\Delta t_k^{(f)} + \Delta t_k^{(R)}\right] \sum_{i=1}^{nf_w} (1) + w$$
$$=> \quad T(w) = [MTTF_k + MTTR_k](nf_w) + w$$

ومنه فإن التوقع الرياضي لزمن انتهاء التنفيذ يعطى بالعلاقة:

$$E(T(w)) = \mathrm{E}([MTTF_k + MTTR_k](nf_w) + w)$$
$$=> \qquad E(T(w)) = [MTTF_k + MTTR_k][E(nf_w)] + w \qquad (3)$$

إذاً من أجل توقع زمن انتهاء تنفيذ التطبيق المكوَّن من w وحدة عمل يكفي توقع عدد مرات حدوث الخطأ nf_w، حيث يمكن النظر إليه على أنّه متغير عشوائي منقطع يتبع توزيعاً هندسياً بوسيط p يعبّر عن احتمال النجاح، ومنه فإنّ العلاقة $nf_w = n$ تكافئ الحدث التالي:

$$\Delta t_1^f < w, \ \ \Delta t_2^f < w, \ldots\ldots, \ \ \Delta t_n^f < w, \ \ \Delta t_{n+1}^f \geq w$$
$$\Delta t_l^f = t_l - t_{l-1} \qquad \text{حيث}$$

إذاً يتحقق النجاح عندما Δt_{n+1}^f يبلغ حده الأدنى وهو w وبالتالي فإن الوسيط p في هذه الحالة يساوي احتمال البقاء في الحالة 1 في الفترة الزمنية $\Delta t_{n+1}^f = w$.

ومنه: $E(n_w) = \frac{1}{p} = \frac{1}{p_1(w)}$

لحساب الاحتمال $p_1(w)$ نستخدم معادلات كولموغروف التقدمية(Kolmogorov forward equations) التي تعرّف تغيّر احتمال الانتقال من الحالة i إلى الحالة j خلال الفترة t بالشكل:

$p'_{ij}(t) = \sum_{k=0}^{r} a_{kj} p_{ik}$ أو بالشكل المصفوفي : $P'(t) = P(t).A$.

تعبّر A عن مصفوفة معدلات الانتقال بين حالات ماركوف[9].

فنحصل على المعادلتين التفاضليتين العاديتين :

$$\left.\begin{array}{l} -\mu p_0(t) + \lambda p_1(t) = p'_0(t) \\ \mu p_0(t) - \lambda p_1(t) = p'_1(t) \end{array}\right] *$$

وفقاً للشروط الابتدائية[3]: $p_0(0) = 0\,, p_1(0) = 1$

[3] **يتم عادةً استخدام معادلات الحالة وذلك عندما نعلم أن المكون في لحظة البدء ولتكن t=0 كان في الحالة i عندها للسهولة نكتب $p_j(t)$ على الشكل $p_{ij}(t)$ [9].**

بحل هذه المعادلات[9] نجد أنّ :

$$E(nf_w) = \frac{1}{p_1(w)} = \frac{\mu + \lambda}{\mu + \lambda e^{-(\mu+\lambda)w}}$$

بالتعويض في العلاقة (3) نجد:

$$E\big(T(w)\big) = [MTTF_k + MTTR_k]\left(\frac{\mu_k + \lambda_k}{\mu_k + \lambda_k e^{-(\mu_k+\lambda_k)w}}\right) + w \quad (4)$$

هذه العلاقة توضّح زمن تنفيذ تطبيق متوازي على مكوّن واحد فقط، ولجعلها تمثل زمن التنفيذ على بيئة تفرّعية مؤلفة من p معالج مثلاً نقوم باستبدال $MTTF_k , MTTR_k$ بـ $MTTF(p), MTTR(p)$، ونقوم بضرب المقدار $p_1^k(w)$ (احتمال بقاء المعالج رقم k في الحالة رقم 1 خلال الفترة w) بعدد المعالجات الكلّي p حيث أنّ احتمال نجاح العمل والبقاء في الحالة 1 طيلة فترة التنفيذ يجب أن يشمل كافّة المعالجات.

لتصبح العلاقة (4) بالشكل:

$$E\big(T(w)\big) = [MTTF(p) + MTTR(p)]\left(\frac{1}{p * p_1^k(w)}\right) + w \;\; ; k = 1, \dots, p$$

أو نكتبها بالشكل:

$$E\big(T(w)\big) = [MTTF(p) + MTTR(p)]\left(\frac{1}{p} * \frac{\mu_k + \lambda_k}{\mu_k + \lambda_k e^{-(\mu_k+\lambda_k)w}}\right) + w$$
$$; k = 1, \dots, p$$

لتعيين كلٍّ من $MTTF(p), MTTR(p)$ نفرض بأنّ زمن الوصول إلى الفشل لمعالجات البيئة تملك جميعها التوزيع الاحتمالي نفسه وبنسب متساوية λ (iid(identical and independent distributed، وكذلك الحال بالنسبة للزمن اللازم لإعادة الإصلاح، وأنّه في مرحلة الإصلاح يتم تجديد المعالج الفاشل فقط بينما تكون المعالجات الأخرى في حالة خمول(idle) تنتظر ريثما يتم الاسترجاع إلى نقطة الاستعادة العامّة المخزّنة، وبالتالي في حال فرضنا أنّ جميع المعالجات تملك نفس التوزيع الاحتمالي لزمن الإصلاح وبمعدلات متساوية عندها يكون الزمن اللازم لإصلاح النظام المكون من p معالج مساوياً للزمن اللازم لإصلاح مكون واحد أي:

$$MTTF(p) = \frac{MTTF_k}{p} = \frac{1}{p\lambda}$$

$$MTTR(p) = MTTR_k = \frac{1}{\mu}$$

$$\mu_i = \mu_j = \mu \quad , \quad \lambda_i = \lambda_j = \lambda \quad \forall i, j \in \{1, \dots, p\} \; and \; i \neq j$$

ومنه نكتب الشكل النهائي لتوقع زمن تنفيذ تطبيق متوازي في بيئة موزعة مؤلفة من p معالج ومن دون استخدام آلية للتسامح مع الأعطال بالشكل:

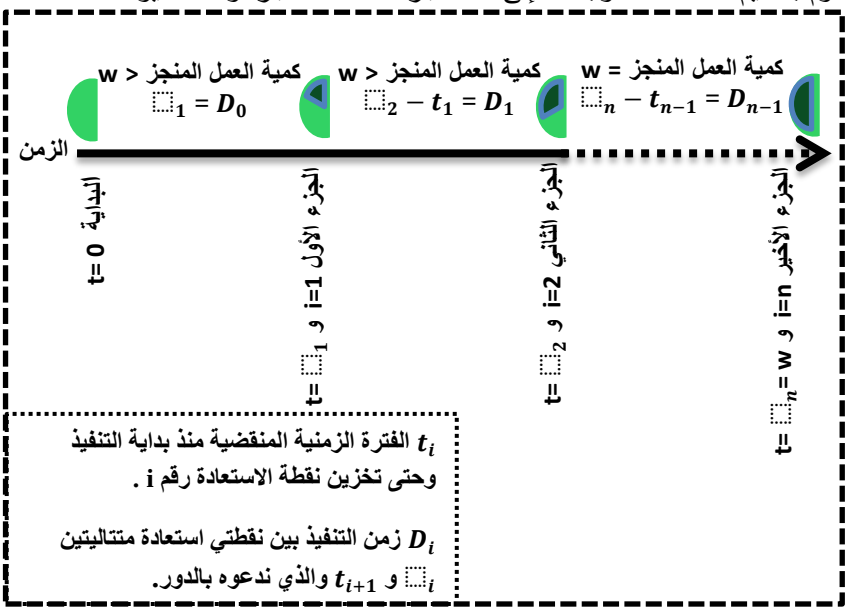
$$E\big(T(w)\big) = \left[\frac{1}{p\lambda} + \frac{1}{\mu}\right]\left(\frac{1}{p} * \frac{\mu + \lambda}{\mu + \lambda e^{-(\mu+\lambda)w}}\right) + w \quad (5)$$

نلاحظ أنّ زمن انتهاء تنفيذ التطبيق عند ثبات كل من MTTF & MTTR يتزايد عندما يزداد عدد المهام w وذلك يعود سببه بشكل أساسي إلى الحد الأسي $\left(e^{(\mu+\lambda)w}\right)$ والذي يمثّل عدد الأعطال المتوقع حدوثها، فعند زيادة هذا المقدار سيزيد العبء على التطبيق في عدد مرات إعادة التنفيذ نتيجة حدوث الأخطاء؛ يزداد هذا المقدار بشكل رئيسي

عندما يكون حجم العمل المتوازي w كبير نسبياً، إذاً لتخفيض تلك الزيادة علينا تقسيم العمل المتوازي إلى أجزاء صغيرة ومستقلة.

3.3. النموذج الرياضي لزمن انتهاء تنفيذ التطبيق باستخدام آلية للتسامح مع الأعطال:

لتفادي إعادة تنفيذ المهام التي تم إنجازها قبل العطل نستخدم نقاط الاستعادة المتناسقة لحفظ تنفيذ هذه المهام ومن ثم استرجاعها بعد العطل، يتم توزيع أزمنة أخذ نقاط الاستعادة وفق خوارزمية جدولة ما ولنرمز لها بالرمز π، تتألف من n نقطة استعادة يتعلق عددها بكمية العمل المطلوب تنفيذه w، إذاً وفقاً لهذا التعريف فإنّ خوارزمية جدولة نقاط الاستعادة سوف تقوم بتقسيم العمل المطلوب w إلى عدة أجزاء عددها n جزء وهذا ما يوضحه الشكل (2).



الشكل 2: استراتيجية جدولة نقاط الاستعادة π

يتضح لنا من الشكل(2) أعلاه أنّ كل جزء D_i يمكن أن نطبّق عليه العلاقة (5) لحساب توقع انتهاء تنفيذه، فكل جزء هو عبارة عن تطبيق جزئي زمن تنفيذه هو زمن تنفيذ جزء العمل المُنجَز بين نقطتي استعادة (بوجود أخطاء وبدون آلية للتسامح مع الأعطال) مضافاً له زمن أخذ نقطة الاستعادة أي: $D_i = T(w_i) + c_i$ أمّا لحساب توقع عدد مرات إعادة التنفيذ لكل فترة فنقوم بإعادة حل جملة المعادلات (*) مع الشروط الابتدائية(لحظة البدء) الخاصة بكل فترة لأن الحل وحيد وخاص.

فنجد أن الحل المتمثل بالمقدارين $p_1(t), p_0(t)$ يكون على الشكل التالي:

$$p_0(t) = \frac{\lambda}{\mu+\lambda} - \frac{\lambda}{\mu+\lambda} e^{(\mu+\lambda)(t_i - t)}$$

$$p_1(t) = \frac{\mu}{\mu+\lambda} + \frac{\lambda}{\mu+\lambda} e^{(\mu+\lambda)(t_i - t)} \qquad ; t \in [t_i, t_{i+1}]$$

حتى يتحقق النجاح ويتم إنجاز كمية من العمل قدرها $_i$ فيجب أن تأخذ الفترة الزمنية t القيمة الآتية: $t =$ $t_i + w_i + c_i$ ونلاحظ هنا أننا قد أضفنا زمن تخزين نقطة الاستعادة، فلن يحدث النجاح بشكل حقيقي إلّا إذا تم تخزين العمل المنجز بنجاح.

ومنه فإنّ التوقع الرياضي لزمن تنفيذ العمل الجزئي w_i ضمن بيئة تفرعية مؤلفة من p معالج تعطى بالعلاقة :

$$E\big(T_i(w_i)\big) = [MTTF(p) + MTTR(p)]\left(\frac{1}{p * p_1(t_i + w_i + c_i)}\right) + w_i$$

حيث :

$$p_1(t_i + w_i + c_i) = \frac{\mu}{\mu+\lambda} + \frac{\lambda}{\mu+\lambda} e^{-(\mu+\lambda)(w_i+c_i)} \quad ; i = 1,..,n$$

نشير إلى أنّ c_i تمثل كلفة تسجيل نقطة الاستعادة في الفترة i، حيث تتعلق هذه الكلفة بكمية العمل المخزّن في تلك الفترة، إذاً من الممكن أن نكتب العلاقة التالية: $c_i = \beta w_i$ (تشير β إلى أكبر كلفة ممكنة لتخزين مهمة ما في التطبيق).

إنّ زمن انتهاء تنفيذ التطبيق المتوازي المكون من w وحدة عمل ضمن البيئة الموزعة المدروسة هو مجموع أزمنة انتهاء تنفيذ الأجزاء الخاصة بكل قسم للتطبيق i مضافاً له كلفة تسجيل نقاط الاستعادة خلال التنفيذ أي:

$$T(w) = \sum_{i=1}^{n} D_i = \sum_{i=1}^{n} (T(w_i) + c_i)$$

فالتوقع الرياضي لهذا المقدار يعطى بالعلاقة:

$$E\big(T(w)\big) = \sum_{i=1}^{n} \big[E\big(T_i(w_i)\big) + c_i\big]$$

إذاً فإنّ مسألة الحل الأمثل التي نبحث عنها هي من الشكل :

$$\min \; E(T(w))$$

حيث:

$$E\big(T(w)\big) = \sum_{i=1}^{n} \left[\left(\frac{\frac{1}{p\lambda} + \frac{1}{\mu}}{p} \right) \frac{\mu+\lambda}{\mu + \lambda e^{-(\mu+\lambda)(w_i+\beta w_i)}} + w_i + c_i \right]$$

ممكن أن نرمز للمقدار الثابت $(\mu+\lambda)(\frac{1}{\mu p} + \frac{1}{{}_2\lambda})$ بالرمز A وكذلك لـ $(\mu+\lambda)(1+\beta)$ بالرمز α فيصبح المجموع بالشكل:

$$E\big(T(w)\big) = \sum_{i=1}^{n} \left[\frac{A}{\mu + \lambda e^{-\alpha w_i}} + w_i + c_i \right] \qquad (6)$$

وفقاً للشروط التالية:

$$w_i \le w \,,\; w_i > 0 \quad \forall i = 1, \dots, n$$

أو بشكل آخر:

$$E\big(T(w)\big) = \sum_{i=1}^{n} \left[\frac{A}{\mu + \lambda e^{-\alpha w_i}} \right] + \sum_{i=1}^{n} (w_i + \beta w_i)$$

$$=> \qquad E\big(T(w)\big) = \sum_{i=1}^{n} \left[\frac{A}{\mu + \lambda e^{-\alpha w_i}} \right] + (1+\beta)w \qquad (7)$$

مع الأخذ بعين الاعتبار أنّ:

- p عدد المعالجات.
- n عدد نقاط الاستعادة التي سيتم تخزينها للتطبيق.
- $w = \frac{w}{p} + \gamma w$ وذلك حسب قانون آمدال (Amdahl low)[10]، يشير γ إلى نسبة العمل التسلسلي في التطبيق المتوازي.
- $\sum_{i=1}^{n} w_i = w$.

2) حل النموذج:

حصلنا فيما سبق على نموذج لمتوسط زمن انتهاء تنفيذ التطبيق المتوازي في بيئة الحوسبة الشبكية(7) وباستخدام آلية نقاط استعادة متناسقة للتسامح مع الأعطال، ووجدنا أنّ النموذج كان عبارة عن مسألة بحوث عمليات تبحث عن قيم متغيرات النموذج من أجل الحصول على أصغر قيمة ممكنة لزمن تنفيذ التطبيق.

يتمثّل الحل بإيجاد القيم المناسبة لمركبات الشعاع $(w_1, w_2, \dots, w_n)$ حتى يكون التوقع السابق(7) أصغر ما يمكن، بطريقة أخرى فإننا نبحث عن أفضل عدد لنقاط الاستعادة من أجل أن نحقق توازن بين كلفة تسجيل نقاط الاستعادة من جهة، والخسارة الناتجة عن إعادة تنفيذ العمل الذي لم يتم تخزينه قبل حدوث العطل من جهة أخرى، وبالتالي فإنّ الحدّ المؤثر في علاقة التوقع السابقة(7) هو $\sum_{i=1}^{n}\left[\frac{A}{\mu+\lambda e^{-\alpha w_i}}\right]$ والذي يمثل عدد المرات المتوقعة لإعادة تنفيذ العمل المكون من w_i وحدة عمل.

فإذا تمّ اختيار قيم مناسبة لـ w_i سيؤدي ذلك إلى الحصول على أصغر زمن تنفيذ ممكن للتطبيق، مما يعني أننا نبحث عن الحل الأمثل للمسألة الجديدة:

$$\min \sum_{i=1}^{n}\left[\frac{A}{\mu + \lambda e^{-\alpha w_i}}\right] \qquad (8)$$

وفقاً للشروط:

$$w_i \le w\ ,\ w_i > 0 \quad \forall i = 1, \dots, n$$

1.4. قابلية الحل:

نلاحظ من شكل المسألة(8) أننا قد حصلنا على التابع التالي:

$$f: D \subseteq R^n \to R : X \to f(X)$$

$$f(X) = \sum_{i=1}^{n}\left[\frac{A}{\mu + \lambda e^{-\alpha(w_i)}}\right] \qquad ; X = (w_1, w_2, \dots, w_n)$$

نسمي D بالمجموعة الملائمة للتابع feasible set ونكتبها بالشكل:

$$D = \{X \in R^n \mid\ a^T X \le w\ \wedge\ a^T X \ge \varepsilon\}$$
$$; w, \varepsilon \in R\ \wedge\ a \in R^n\ \wedge\ a_i = 1;\ \forall i = 1, \dots, n$$

وهي مجموعة محدّبة[11]، حيث ε هو الحد الأدنى الممكن لكل فترة عمل $_i$.

بفرض أنّ: $g(\ \ _i) = \mu + \lambda e^{-\alpha w_i} \overset{\text{ومنه}}{\Rightarrow} g'(w_i) = -\lambda\alpha e^{-\alpha w_i}$

$$\overset{\text{ومنه}}{\Rightarrow} \quad g''(w_i) = +\lambda\alpha^2 e^{-\alpha w_i} > 0$$

إذاً الدالة $g(w_i)$ محدّبة تماماً (strictly convex function)، ولدينا مقلوب دالة محدّبة هو دالة محدّبة ومجموع دوال محدبة هو دالة محدّبة[11] ومنه فإنّ دالة الهدف $f(X)$ دالة محدّبة.

بالتالي فإن هناك قيمة صغرى تبلغها الدالة $f(X)$ وهي قيمة صغرى شاملة (في الدوال المحدبة كل قيمة صغرى محلّيّاً هي قيمة صغرى شاملة)[11].

تؤول المسألة إلى مسألة إيجاد قيمة صغرى لتابع محدّب convex optimization problem .

2.4. إيجاد الحل:

بما أنّ التابع الهدف للمسألة محدّب عندها فإنّ المسألة تملك إمّا حلّاً أصغرياً داخلياً أو حدّيّاً وذلك يعتمد على شعاع التدرج للتابع[10] الهدف gradient vector $\nabla f(X)$ والذي يعطى بالعلاقة:

$$\nabla f(X) = \begin{bmatrix} \frac{\partial f}{\partial w_1} \\ \frac{\partial f}{\partial w_2} \\ \vdots \\ \frac{\partial f}{\partial w_n} \end{bmatrix}$$

كل مركبة من مركبات شعاع التدرج تأخذ الشكل:

$$\frac{\partial f}{\partial w_i} = \frac{A\lambda\alpha e^{-\alpha w_i}}{(\mu + \lambda e^{-\alpha w_i})^2} \qquad (9)$$

نلاحظ أنّ جميع مركبات شعاع التدرج موجبة تماماً ولا يوجد قيمة ممكنة لـ w_i تجعل البسط في العلاقة(9) معدوم (ممكن أن ينعدم البسط بقيم معينة للوسطاء A, λ, α ولكننا نبحث عن القيمة الحرجة لمتغيرات النموذج w_i والتي تعدم المشتقات الجزئية)، وبالتالي فالتابع متزايد تماماً بالنسبة لجميع مركباته[11] ويبلغ قيمته الصغرى عند الحد الأدنى لكل مركبة من مركبات الشعاع X من المنطقة الملائمة D.

أي أنّ الحل الأمثل يتحقق عندما تكون جميع مركبات الشعاع X متساوية وتساوي $\varepsilon = \frac{W}{n}$ ، إذاً نتجه لإيجاد أفضل قيمة لـ n تجعل الدالة التالية أصغر ما يمكن:

$$\varphi(n) = n\left(\frac{A}{\mu + \lambda e^{-\alpha\left(\frac{W}{n}\right)}}\right) \qquad (10)$$

حيث نتجت العلاقة(10) من تعويض كل $_i$ في العلاقة(8) بالحد الأدنى لها وهو $\frac{W}{n}$.

بنفس الطريقة السابقة نقوم باشتقاق $\varphi(n)$ بالنسبة لـ n لإيجاد القيمة الحرجة للدالة، فنجد :

$$\varphi'(n) = \frac{A\left(\mu + \lambda e^{-\alpha\left(\frac{W}{n}\right)}\right) - \lambda A\alpha\frac{W}{n}e^{-\alpha\left(\frac{W}{n}\right)}}{\left(\mu + \lambda e^{-\alpha\left(\frac{W}{n}\right)}\right)^2}$$

لنفرض أنّ n_0 هي القيمة الأمثل للنموذج السابق عندها:

$$\varphi'(n_0) = 0$$

ومنه:

$$A\left(\mu + \lambda e^{-\alpha\left(\frac{W}{n_0}\right)}\right) - \lambda A\alpha\frac{W}{n_0}e^{-\alpha\left(\frac{W}{n_0}\right)} = 0$$

$$\mu + \lambda\left(1 - \frac{\alpha w}{n_0}\right)e^{-\alpha\left(\frac{w}{n_0}\right)} = 0$$

بفرض أنّ $y = 1 - \frac{\alpha w}{n_0}$ عندها فإنّ $e^{-\alpha\frac{w}{n_0}} = e^{y-1}$ أي:

$$\mu + \lambda y e^{y} e^{-1} = 0$$

$$y e^{y} = -\frac{\mu}{\lambda}e^{-1}$$

حل هذه المعادلة هو[11]:

$$y = \mathbb{W}\left(-\frac{\mu}{\lambda}e^{-1}\right)$$

حيث $\mathbb{W}$ هي دالة لامبرت (Lambert function)، إذاً فالقيمة الأمثل لـ n والتي تجعل زمن تنفيذ التطبيق أصغر ما يمكن هي:

$$n_0 = \frac{\alpha w}{1 - \mathbb{W}\left(-\frac{\mu}{\lambda}e^{-1}\right)} \qquad (11)$$

3.4. مناقشة الحل:

- نستنتج مما سبق بأنّ الحل الأمثل الذي حصلنا عليه من النموذج الرياضي المقترح للبيئة المدروسة يبين أن نقاط الاستعادة **الدورية** ستحقق أفضل زمن تنفيذ ممكن للتطبيق، وهذا يمكن توضيحه بشكل واقعي كما يلي: تمر مكونات بيئة الحوسبة الشبكية بثلاث حالات أساسية وهي مرحلة الاكتشاف المبكر للأخطاء ومرحلة العمل المفيد وأخيراً مرحلة التآكل والتي تبدي معدل مرتفع لظهور الأخطاء[12]، بفرض أن أعطال الأجهزة تتبع للتوزيع الأسي فهذا يعني أنها تقضي معظم وقتها في العمل المفيد، حيث تكون معدلات حدوث أعطالها شبه ثابتة طيلة فترة التنفيذ، مما يعني أن اختيار كمية ثابتة لحجم العمل في كل فترة يبدو مفيداً، أمّا حالتي الاضطراب في البداية والنهاية فتكونان على فترات قصيرة بالنسبة لمرحلة العمل المفيد، مما يعني أنّ الكلفة الزائدة لنقاط الاستعادة في بداية التنفيذ سوف تتوازن مع كمية العمل الضائع في المراحل الأخيرة من العمل بشكل تقريبي.
- كذلك يمكن لنا أن نرى أن نقاط الاستعادة المتناسقة الدورية تقسم سير العمل المتوازي إلى مراحل مستقلة ومتساوية، مما يجعل زمن الخسارة (إعادة تنفيذ جزء العمل غير المخزن بعد العطل) في كل جزء متساوٍ تقريباً، وهذا يؤدي وضوحاً إلى أقل خسارة ممكنة وبالتالي أقل زمن تنفيذ ممكن للتطبيق المتوازي المدروس.
- من العلاقة (11) يمكن أن نستنتج ما يلي:

بما أنّ $n_0 > 0$ إذاً $n_0 < \alpha w$ حيث أن $\alpha = (\mu + \lambda)(1 + \beta)$ ومنه ممكن أن نكتب $n_0 = \lfloor \alpha w \rfloor$ وبالتالي عند ثبات w وازدياد (تناقص) معدل حدوث الأعطال λ في التطبيق فسوف تزداد (تنقص) قيمة n_0 أيضاً أي العدد الأمثل لنقاط الاستعادة للتطبيق، كذلك بثبات α وازدياد (تناقص) قيمة العمل المتوازي المُقدَّر w سوف تزداد (تنقص) عدد نقاط الاستعادة التي يحتاجها التطبيق لتحقيق أقل زمن تنفيذ ممكن.

الاستنتاجات والتوصيات:

عرضنا في هذا البحث مشكلة الكلفة الزمنية المضافة إلى بيئة الحوسبة الشبكية عند استخدام آلية للتسامح مع الأعطال تعتمد على التخزين المتكرر لحالة التطبيق المتوازي ومن ثم الاسترجاع بعد العطل، فقمنا بتقديم دراسة تحليلية لأداء

هذه الآلية ضمن المعايير التي تفرضها البيئة، وذلك باستخدام نموذج ماركوف لتمثيل نوع العطل المدروس، ومعادلات كولموغروف التقدمية للتعبير عن احتمال تغيّر حالة المكون بين التوقف والعمل خلال الزمن، وتوصلنا من خلالها إلى نموذج رياضي يحدد لنا زمن انتهاء تنفيذ التطبيق المتوازي بدون ومع آلية للتسامح مع الأعطال، بحل هذا النموذج استطعنا تحديد عدد نقاط الاستعادة الأمثل للحصول على أفضل زمن تنفيذ للتطبيق، ووجدنا أن التقسيم الأمثل للتنفيذ يكون عندما تتساوى أجزاء التطبيق المُنفّذة بين نقاط الاستعادة.

المراجع

المراجع الأجنبية:

1) TROBEC, R.,VAJTERSIC M., and ZINTERHOF, P. *Parallel Computing – Numerics, Applications, and Trends.***,** in Springer-Verlag, London, 2009**.**

2) AVIZIENIS, A. , LAPRIE, J-C. and RANDELL, B. *Dependability and its threats - a taxonomy*, in IFIP Congress Topical Sessions, 2004, pages: 91–120.

3) DABROWSKI, C. *Reliability in grid computing systems*, in journal Concurrency and Computation: Practice & Experience - A Special Issue from the Open Grid Forum, Vol. 21 Issue 8, 2009, Pages: 927-959.

4) AHMED W., HASAN O., and TAHAR S. *Formal Dependability Modeling and Analysis: A Survey*. CICM: International Conference on Intelligent Computer Mathematics, 2016, Pages: 132-147.

5) LIU,Y., NASSAR,R., LEANGSUKSUN,C., NAKSINEHABOON,N., PAUN,M., AND SCOTT,S. *An Optimal Checkpoint/Restart Model for a Large Scale High Performance Computing System*, In IPDPS 2008, IEEE,2008 pages 1–9.

6) ÖZSU T. VALDURIEZ P., *Principles of Distributed Database Systems*, Springer New York Dordrecht Heidelberg London, 3rd Edition,2011, 866 pages.

7) RICKY W. BUHER and SALLY C.JOHNSON. *Techniques for Modeling the Reliability of Fault-Tolerant Systems With the Markov State-Space Approach*. NTRS: NASA Technical Reports Server ,1995, 131 pages.

8) BESSERON,X., JAFAR,S., GAUTIER,T. AND ROCH,J.L. *CCK: An Improved Coordinated Checkpoint/Rollback Protocol for Dataflow Applications in KAAPI*, in: 2nd International Conference on Information & Communication Technologies, vol.2,Issue 11,2006, pages: 3353 – 3358.

9) RAUSAND M. *System Reliability Theory*. John Wiley & Sons, Inc., 2nd Edition, Canada, 2004, 664 pages.

10) AMDAHL,G. *The validity of the single processor approach to achieving large scale computing capabilities*. In AFIPS Conference Proceedings, vol.30,1967, pages 483–485.

11) BOYD,S. AND VANDENBERGHE,L. *Convex Optimization* ,1st Edition,in Cambridge University Press,2004.

12) KLUTKE,G., KIESSLER,P.C AND WORTMAN M. A. *A Critical Look at the Bathtub Curve.* In IEEE transactions on reliability, vol. 52, no. 1, 2003, Pages: 125 – 129.

المراجع العربية:

9) جعفر،سمير. *نموذج رياضي مستمر للتسامح مع الأعطال في التطبيقات المتوازية*، مجلة جامعة البعث، المجلد 38، العدد 13،2016، الصفحات 79-98.