Tolérance aux fautes

Stéphane Devismes

Université Grenoble Alpes

17 septembre 2019

Plan

Plan

Défaillances, erreurs et fautes

On peut résumer les relations entre ces différents termes par :

≪ une faute provoque une erreur qui entraîne une défaillance ≫.

Défaillances/pannes

On parle de la défaillance d'un composant (ou d'un système) lorsque son comportement n'est plus conforme à sa spécification.

Composant = lien de communication ou nœud

nœud = processus, machine ...

Défaillances/pannes

On parle de la défaillance d'un composant (ou d'un système) lorsque son comportement n'est plus conforme à sa spécification.

Composant = lien de communication ou nœud

nœud = processus, machine ...

On parlera de nœud correct (resp. lien fiable) lorsque le nœud (resp. le lien) ne subit pas de défaillance.

Défaillances/pannes

On parle de la défaillance d'un composant (ou d'un système) lorsque son comportement n'est plus conforme à sa spécification.

Composant = lien de communication ou nœud

nœud = processus, machine ...

On parlera de nœud correct (resp. lien fiable) lorsque le nœud (resp. le lien) ne subit pas de défaillance.

Exemples :

- Un **processus** arrête d'exécuter un programme.
- Un **lien de communication** perd un message.

Lien fiable

La spécification d'un lien fiable consiste en la conjonction des trois propriétés suivantes :

Pas de création : Tout message reçu par un processus p venant du processus q a été envoyer au préalable par q à p.

Pas de duplication : Tout message est reçu au plus une fois.

Pas de perte : Tout message envoyé est livré au récepteur en temps fini.

Erreurs

Une erreur est un état du système à partir duquel la poursuite de l'exécution est susceptible de conduire à une défaillance.

Erreurs

Une erreur est un état du système à partir duquel la poursuite de l'exécution est susceptible de conduire à une défaillance.

Des exemples de telles situations sont :

- un chaînage d'une liste chainée corrompu ou un pointeur non initialisé : il s'agit ici d'erreurs logicielles ;
- un câble du réseau déconnecté ou une unité disque éteinte : il s'agit ici d'erreurs matérielles.

Fautes

Une faute est un événement ayant entrainé une erreur.

Fautes

Une faute est un événement ayant entrainé une erreur.

Il peut s'agir

- d'une faute de programmation (pour les erreurs logicielles)

Fautes

Une faute est un événement ayant entrainé une erreur.

Il peut s'agir

- d'une faute de programmation (pour les erreurs logicielles) ou
- d'événements physiques, *e.g.*, usure, malveillance, catastrophe, ... (dans le cas d'erreurs matérielles).

Fautes

Une faute est un événement ayant entrainé une erreur.

Il peut s'agir

- d'une faute de programmation (pour les erreurs logicielles) ou
- d'événements physiques, *e.g.*, usure, malveillance, catastrophe, ... (dans le cas d'erreurs matérielles).

Dans le cadre de ce cours, nous ne considérerons que des fautes matérielles, *e.g.*, coupure d'un lien, perturbation électro-magnétique du signal dans un lien . . .

Faute, erreur, défaillance

La différence est subtile !

Faute, erreur, défaillance

La différence est subtile !

Dans la suite de ce cours, ces trois termes seront considérés comme synonymes.

Fautes dans les réseaux

Les réseaux sont naturellement sujets aux fautes.

Fautes dans les réseaux

Les réseaux sont naturellement sujets aux fautes.

Plus le nombre de processus est important, plus la probabilité qu'un composant du réseau subisse une faute durant l'exécution d'un protocole est importante.

Fautes dans les réseaux

Les réseaux sont naturellement sujets aux fautes.

Plus le nombre de processus est important, plus la probabilité qu'un composant du réseau subisse une faute durant l'exécution d'un protocole est importante.

Par exemple, si on considère le réseau internet (350 millions de serveurs en 2006), il est impossible d'imaginer qu'un tel réseau puisse fonctionner ne serait-ce qu'une heure sans subir la moindre faute !

Fautes dans les réseaux

Les réseaux sont naturellement sujets aux fautes.

Plus le nombre de processus est important, plus la probabilité qu'un composant du réseau subisse une faute durant l'exécution d'un protocole est importante.

Par exemple, si on considère le réseau internet (350 millions de serveurs en 2006), il est impossible d'imaginer qu'un tel réseau puisse fonctionner ne serait-ce qu'une heure sans subir la moindre faute !

Il faut aussi noter que la plupart des réseaux actuels sont constitués de machines ≪ grand public ≫ produites en grand nombre à prix réduit : d'où, un risque de défaut plus important.

Plan

Critères pour la classification des fautes

Les fautes sont généralement classées suivant différents critères :

- L'origine de la faute.
- La cause de la faute.
- La durée de la faute.
- La détectabilité de la faute.

L'origine de la faute

Le type de composant qui est responsable de la faute : lien de communication ou processus.

La cause de la faute

La faute peut être

- bénigne : non volontaire, *e.g.*, due à un problème matériel, ou
- maligne : due à une intention (malveillante ou malicieuse) extérieure au système.

La durée de la faute

- Si la durée d'une faute est supérieure au temps restant de l'exécution du protocole, elle est dite définitive ou franche,
- Sinon elle est dite transitoire ou intermittente.

La durée de la faute

- Si la durée d'une faute est supérieure au temps restant de l'exécution du protocole, elle est dite définitive ou franche,
- Sinon elle est dite transitoire ou intermittente.

La différence entre transitoire et intermittente est définie par la fréquence.

- Dans le premier cas, elle se produit de manière isolée (en moyenne une fois pendant le temps d'exécution du protocole).
- Dans le second cas, elle se produit plusieurs fois (en moyenne, plusieurs fois pendant le temps d'exécution du protocole).

La détectabilité de la faute

Une faute est détectable si son incidence sur la cohérence de l'état d'un processus permet à celui-ci de s'en apercevoir.

Plan

Panne crash

Panne franche d'un processus.

Le processus cesse définitivement de faire des pas de calculs.

Panne crash

Panne franche d'un processus.

Le processus cesse définitivement de faire des pas de calculs.

Perte intermittente de messages

Régulièrement un lien perd des messages.

Perte intermittente de messages

Régulièrement un lien perd des messages.

Deux hypothèses sont généralement utilisées dans ce cas :

- Soit on suppose que les pertes sont équitables,
- Soit on suppose qu'il y a un taux de pertes de message (connu ou inconnu des processus).

Perte intermittente de messages

Régulièrement un lien perd des messages.

Deux hypothèses sont généralement utilisées dans ce cas :

- Soit on suppose que les pertes sont équitables,
- Soit on suppose qu'il y a un taux de pertes de message (connu ou inconnu des processus).

Lorsque les pertes sont équitables, le lien de commmunication vérifie l'hypothèse suivante : si des messages sont envoyés infiniment souvent, alors une infinité de messages est livrée.

Perte intermittente de messages

Régulièrement un lien perd des messages.

Deux hypothèses sont généralement utilisées dans ce cas :

- Soit on suppose que les pertes sont équitables,
- Soit on suppose qu'il y a un taux de pertes de message (connu ou inconnu des processus).

Lorsque les pertes sont équitables, le lien de commmunication vérifie l'hypothèse suivante : si des messages sont envoyés infiniment souvent, alors une infinité de messages est livrée.

On parle aussi de fautes par omission : à divers instants de l'exécution, un composant du réseau omet de communiquer avec un autre en réception ou en émission.

Faute transitoire

C’est un comportement erroné d’un à plusieurs composants du réseau durant une certaine période (finie).

Une fois que cette période est passée, les composants reprennent un comportement correct. Cependant, l’état du système est perturbé.

Faute transitoire

C'est un comportement erroné d'un à plusieurs composants du réseau durant une certaine période (finie).

Une fois que cette période est passée, les composants reprennent un comportement correct. Cependant, l'état du système est perturbé.

Le système **subit les effets de la faute**, *e.g.*, certains messages ont été corrompus.

Faute transitoire

C'est un comportement erroné d'un à plusieurs composants du réseau durant une certaine période (finie).

Une fois que cette période est passée, les composants reprennent un comportement correct. Cependant, l'état du système est perturbé.

Le système **subit les effets de la faute**, *e.g.*, certains messages ont été corrompus.

On parlera alors de fautes d'état : des composants du système ont subi un changement d'état non prévu par l'algorithme.

Par exemple, cela peut être des corruptions de mémoires locales de processus ou de contenus de message, ou encore de la duplication de messages.

Fautes byzantines

Les fautes byzantines sont dues à des processus byzantins qui ont un comportement arbitraire, ne suivant plus (nécessairement) le code de leurs algorithmes locaux.

Fautes byzantines

Les fautes byzantines sont dues à des processus byzantins qui ont un comportement arbitraire, ne suivant plus (nécessairement) le code de leurs algorithmes locaux.

Cela peut être dû à une erreur matérielle, un virus ou la corruption du code de l'algorithme.

Plan

Idée

Les réseaux modernes sont à grande-échelle et fait de machines hétérogènes et produites en masses à faible coût, *e.g.*

Idée

Les réseaux modernes sont à grande-échelle et fait de machines hétérogènes et produites en masses à faible coût, *e.g.*

- **Internet**
  - 17,6 milliard d'objets connectés en 2016
  - Internet des objets

Idée

Les réseaux modernes sont à grande-échelle et fait de machines hétérogènes et produites en masses à faible coût, *e.g.*

- **Internet**
  - 17,6 milliard d'objets connectés en 2016
  - Internet des objets
- **Réseaux sans fils**
  - Communication radio : beaucoup de pertes de messages
  - Crash de machines à cause des batteries limitées

Idée

Les réseaux modernes sont à grande-échelle et fait de machines hétérogènes et produites en masses à faible coût, *e.g.*

- **Internet**
  - 17,6 milliard d'objets connectés en 2016
  - Internet des objets
- **Réseaux sans fils**
  - Communication radio : beaucoup de pertes de messages
  - Crash de machines à cause des batteries limitées

$\Rightarrow$ Forte probabilité de pannes

$\Rightarrow$ Intervention humain impossible

Idée

Les réseaux modernes sont à grande-échelle et fait de machines hétérogènes et produites en masses à faible coût, *e.g.*

- **Internet**
  - 17,6 milliard d'objets connectés en 2016
  - Internet des objets
- **Réseaux sans fils**
  - Communication radio : beaucoup de pertes de messages
  - Crash de machines à cause des batteries limitées

$\Rightarrow$ Forte probabilité de pannes

$\Rightarrow$ Intervention humain impossible

$\Rightarrow$ Besoin d'algorithmes distribués tolérant les pannes, *i.e.*, une prise en compte automatique de la possibilité de l'arrivée de pannes au niveau algorithmique.

Objectif

L'objectif principal de la tolérance aux fautes est d'éviter de réinitialiser le réseau après chaque panne.

Ainsi, il faut concevoir des algorithmes capables d'assurer un service ≪ minimum ≫ malgré les pannes.

A priori, un système n'est généralement que partiellement touché par les pannes. Donc, les sites corrects peuvent continuer à fournir un service, de plus ils peuvent prendre en charge les tâches allouées aux sites défaillants.

Il peut en résulter une perte de performance mais pas (nécessairement) de fonctionnement erroné.

Approches pour la tolérance aux fautes

Deux catégories principales :

- Les algorithmes robustes.
- Les algorithmes (auto)stabilisants.

Algorithmes robustes

Ils abordent le problème de tolérance aux fautes selon une approche pessimiste où les processus suspectent toutes les informations qu'ils recoivent.

Algorithmes robustes

Ils abordent le problème de tolérance aux fautes selon une approche pessimiste où les processus suspectent toutes les informations qu'ils recoivent.

Le but est de ≪ masquer ≫ l'effet des pannes à l'utilisateur : on garantit toujours la spécification de l'algorithme en dépit des pannes.

Les algorithmes (auto)stabilisants

Ils abordent le problème selon une approche optimiste, on fait confiance au système mais si on détecte un dysfonctionnement, on le corrige.

Les algorithmes (auto)stabilisants

Ils abordent le problème selon une approche optimiste, on fait confiance au système mais si on détecte un dysfonctionnement, on le corrige.

Le système (en particulier l'utilisateur) subit l'effet des pannes : cela cause un comportement anormal de processus, parfois même non-défaillants, mais garantit le retour vers un comportement global normal en un temps fini après que les fautes ont cessé, c'est une approche ≪ non-masquante ≫.

Les algorithmes (auto)stabilisants

Ils abordent le problème selon une approche optimiste, on fait confiance au système mais si on détecte un dysfonctionnement, on le corrige.

Le système (en particulier l'utilisateur) subit l'effet des pannes : cela cause un comportement anormal de processus, parfois même non-défaillants, mais garantit le retour vers un comportement global normal en un temps fini après que les fautes ont cessé, c'est une approche ≪ non-masquante ≫.

L'autostabilisation est considérée comme lightweight (*i.e.*, à surcoût faible) par rapport à l'approche robuste.