Transferência de Dados entre Grupos de Processos no Modelo BAR

Xavier Vilaça, João Leitão, and Luís Rodrigues

INESC-ID, Instituto Superior Técnico, Universidade Técnica de Lisboa
{xvilaca,jleitao}@gsd.inesc-id.pt, ler@ist.utl.pt

Resumo Este artigo aborda o problema da transferência fiável de informação entre um conjunto de N produtores e um conjunto de N consumidores no modelo de sistema BAR (Bizantinos, Altruístas e Racionais). O algoritmo tolera a presença de, no máximo, f produtores e/ou consumidores Bizantinos, podendo os restantes participantes adoptar um comportamento Racional ou Altruísta. Desde que $N \geq 2f + 1$, garante-se que, no final da transferência, todos os consumidores não Bizantinos possuem os dados correctos. Na presença de um observador confiável, garante-se, também, que todos os produtores e consumidores que executaram correctamente o algoritmo são devidamente recompensados. De forma a demonstrar que é do interesse de qualquer participante Racional executar o algoritmo tal como especificado, prova-se que este é um equilíbrio de Nash.

Abstract This paper addresses the problem of reliably transferring information from a set of N producers to a set of N consumers in the BAR model, i.e., the presence of up to f Byzantine producers and/or consumers, and any number of Rational processes. We assume that $N \geq 2f + 1$. Our algorithm guarantees that at the end of the transfer all correct consumers have the produced data. Furthermore, we assume the existence of a trusted observer: when the transfer terminates, the observer should be provided with enough evidence to testify that the producers and consumers have participated in the transfer. We show that our algorithm is a Nash equilibrium.

1 Introdução

As redes entre-pares têm vindo a revelar-se adequadas para o processamento de grandes volumes de dados através da partilha de recursos entre vários voluntários. Exemplos incluem o projecto *SETI@home*[3] e, mais recentemente, a infraestrutura *Boinc*[2]. Ao estender estes sistemas para suportar modelos de divisão de tarefas como o *map-reduce*, torna-se necessário transferir informação de forma fiável directamente entre voluntários, nomeadamente a informação produzida por *mappers* para ser consumida pelos *reducers*.

Sendo um sistema entre-pares um sistema aberto, convém adoptar um modelo de sistema que capture a grande variedade de comportamentos dos voluntários, o que é possível através do modelo *Byzantine-Altruistic-Rational* (BAR)[1]. Este modelo classifica os participantes em Bizantinos, caso adoptem um comportamento arbitrário, Racionais, se o seu principal objectivo for maximizar uma função utilidade conhecida, e Altruístas, se obedecerem ao comportamento especificado.

Sistemas que não sejam tolerantes ao comportamento Racional podem ser vulneráveis ao fenómeno da *Tragédia dos Comuns*[10], em que os nós Racionais, para minimizarem os custos de participação no sistema, não executam as tarefas de que

estão incumbidos, esperando que esse serviço seja prestado por nós Altruístas. Caso todos os participantes adoptem um comportamento Racional desta natureza, o sistema deixa de providenciar qualquer serviço. Para modelar o comportamento de um sistema com nós Racionais, é possível recorrer à Teoria de Jogos[16], na qual o conjunto de interacções entre nós é visto como um jogo, em que os jogadores correspondem às entidades e as jogadas às interacções (o plano de acção de cada jogador pode incluir várias jogadas, sendo denominado por estratégia). Neste artigo, focamos-nos no conceito de Equilíbrio de Nash, uma situação em que nenhum jogador consegue aumentar a sua utilidade alterando, unilateralmente, a sua estratégia inicial, dadas as estratégias de todos os outros jogadores. Note-se que, tal como em [1,7], neste artigo, os jogadores são considerados avessos ao risco, isto é, não estão dispostos a incorrer em qualquer risco se, de alguma forma, isso puser em causa a sua utilidade.

Este artigo introduz o problema de transferência fiável de dados no modelo BAR, denominado por NBART, em que se considera a existência de N produtores e N consumidores, assumindo que, no máximo, existem f participantes Bizantinos em cada um destes conjuntos. De acordo com o modelo BAR, os restantes participantes são Racionais ou Altruístas. Todos os produtores não Bizantinos possuem a mesma informação, que deve ser transferida para os consumidores. Durante o processo de transferência, cada produtor é responsável por enviar os seus dados a consumidores, garantindo que, no final, todos os consumidores não Bizantinos possuem os dados correctos. De forma a recompensar os nós pela sua participação e, desta forma, a incentivar os participantes Racionais a contribuírem, assume-se a existência de uma entidade confiável, denominada por *observador*. A sua função consiste em reunir informação enviada pelos consumidores, o que lhe permite determinar se cada nó participou correctamente na transferência e, consequentemente, se merece ser recompensado. As funções do observador podem ser realizadas em diferido, após o término da transferência, pelo que o observador não assume um papel activo no processo. Desta forma, múltiplas instâncias do protocolo podem ser arbitradas por um único observador, sem que tal seja um problema para a escalabilidade.

As seguintes contribuições resultam deste trabalho: i) Define-se o problema NBART e as suas principais propriedades. ii) Propõe-se um novo algoritmo que resolve o problema indicado. iii) Demonstra-se a correcção do protocolo, e prova-se que este é um equilíbrio de Nash.

O artigo encontra-se organizado da seguinte forma. Inicialmente, descreve-se algum do trabalho relacionado na Secção 2. Seguidamente, na Secção 3, introduz-se o modelo do sistema e caracteriza-se o problema NBART, enumerando as suas principais propriedades. O protocolo que resolve o problema descrito é apresentado na Secção 4. Segue-se uma avaliação analítica da solução, na Secção 5. Por fim, na Secção 6, conclui-se o artigo.

2 Trabalho Relacionado

O problema da difusão fiável de informação foi extensivamente estudado no contexto da tolerância a faltas Bizantinas[9,11,5]. Apesar de estes protocolos poderem ser executados pelos produtores para divulgar os seus valores pelos consumidores, nenhuma das soluções propostas aborda o problema da presença de nós racionais, tornando-as vulneráveis a este tipo de comportamento[7].

No que respeita à tolerância a faltas Bizantinas, importa, também, realçar o trabalho relacionado com o problema da manipulação de registos replicados, abor-

dado em [14,6,15]. Este problema pode ser descrito como um número indeterminado de clientes que manipula um dado valor, replicado por vários servidores diferentes, através de operações de leitura e escrita. O principal foco das soluções enquadradas neste contexto é o de assegurar a coerência dos valores replicados, não se validando a sua correcção. A contrastar com este facto, no problema NBART, visa-se garantir a transferência fiável de informação de produtores para consumidores, protegendo a sua integridade e validando a sua correcção.

O problema da tolerância a nós Racionais foi estudado no contexto de partilha de conteúdos, especialmente, através da aplicação de mecanismos *tit-for-tat*. Contudo, foi demonstrado[18] que mecanismos como os utilizados no Bittorrent[8] são vulneráveis a comportamento Racional.

O modelo BAR foi, inicialmente, introduzido em [1] e refinado em [7]. Estes artigos propuseram uma arquitectura genérica, que serviria de abordagem base para a construção de qualquer protocolo tolerante ao modelo BAR. Propuseram, também, uma nova solução para o problema de manutenção de uma máquina de estados replicada, usando um protocolo de difusão fiável (TRB). Neste contexto, os autores demonstraram que o protocolo TRB proposto por Dolev e Strong[9] pode ser aumentado com mecanismos ∞ *tit-for-tat*[4], que tornam o protocolo num Equilíbrio de Nash. Pode pensar-se que o problema NBART é facilmente resolvido usando o protocol TRB tolerante ao modelo BAR, por exemplo, requerindo que cada produtor execute uma instância diferente do TRB para divulgar os dados por todos os consumidores. Contudo, mecanismos do tipo ∞ *tit-for-tat* pressupõem que os participantes interagem entre si, repetidamente, em múltiplas instâncias do protocolo, o que não acontece no caso do problema NBART.

O modelo BAR, também, foi aplicado noutros contextos, como o da disseminação de informação baseada em *gossip*, nos artigos [13,12]. O trabalho desses autores pressupõe que a fonte da informação é sempre confiável, portanto, os protocolos propostos não são aplicáveis no problema NBART. Os autores de [17] propuseram um sistema de filtragem de *Spam* tolerante ao modelo BAR, em que os participantes estão organizados numa estrutura em escada, de tal forma que nós no topo da escada estão menos sujeitos a receberem *Spam*. Este facto é usado para incentivar os nós Racionais a filtrarem *Spam* e, consequentemente, a subirem na escada. No entanto, este sistema não valida a origem da informação disseminada nem a sua correcção.

Tanto quanto é do nosso conhecimento, até à data, nenhum trabalho abordou o problema da transferência fiável de informação de produtores para consumidores, tolerando a existência de participantes Bizantinos e Racionais.

3 Modelo do Sistema e Definição do Problema

3.1 Modelo do Sistema

Assume-se que o sistema é síncrono, que os canais de comunicação são fiáveis e que a rede é conexa, ou seja, cada nó consegue comunicar com qualquer outro participante num tempo finito e as mensagens não se perdem caso sejam correctamente enviadas. Pressupõe-se a existência de uma intraestrutura de chave pública, que todos os participantes possuem uma chave pública e conhecem as chaves públicas de todos os participantes.

Nada se assume acerca do comportamento dos nós Bizantinos, a não ser o facto de não serem capazes de quebrar os mecanismos criptográficos usados no protocolo. Já em relação aos participantes Racionais, pressupõe-se que o seu objectivo é a obtenção do valor correcto e a sua certificação perante o observador confiável, de forma a serem recompensados pelo seu contributo. Este objectivo exige que todos

os participantes não Bizantinos se coordenem entre si e enviem informação suficiente ao observador confiável, de forma a avaliar o comportamento de cada um dos participantes. Contudo, dado que os nós incorrem em custos pela sua participação no sistema, considera-se que o seu comportamento seja modelado pela maximização de uma dada função de utilidade, que pesa tanto os benefícios como os custos. Por fim, assume-se que um nó Racional, perante múltiplos comportamentos alternativos com a mesma utilidade, escolhe aquele que obedece ao protocolo especificado.

3.2 Definição do Problema

Apresenta-se agora uma definição mais precisa do problema NBART. Cada produtor correcto p possui uma cópia do valor $val(p)$ a ser transferido. Desta forma, para quaisquer dois produtores p_1 e p_2 não Bizantinos, $val(p_1) = val(p_2)$.

O problema é caracterizado pela seguinte interface. A primitiva $produce(p, v)$ permite ao produtor p iniciar a transferência do valor v. Por sua vez, cada consumidor c invoca a primitiva $consume(c, v)$ com o intuito de entregar o valor v, no final da transferência. Finalmente, a primitiva *certify*(*TO*,*evidence*) é usada pelo observador confiável *TO* para dar início ao processo de avaliação de cada participante, com base na informação *evidence*, reunida a partir dos dados enviados pelos consumidores a *TO*.

Para esse efeito, *TO* invoca as funções booleanas *hasProduced (evidence, p_i)* e *hasAcknowledged(evidence, c_j)*, que recebem a informação *evidence* e um produtor p_i ou um consumidor c_j, respectivamente, retornando *verdadeiro* caso a informação indique que o participante em causa participou de forma correcta na transferência, ou retornando *falso*, caso contrário.

As propriedades do NBART são as seguintes:

- **NBART 1** *(Terminação):* Alguma vez (*eventually*), cada consumidor não Bizantino c entrega o valor correcto v, *i.e*, invoca $consume(c, v)$.
- **NBART 2** *(Validade):* Se um consumidor não Bizantino entrega um valor v, então esse valor foi produzido por um produtor não Bizantino.
- **NBART 3** *(Integridade):* Nenhum consumidor não Bizantino invoca a primitiva *consume* mais do que uma vez.
- **NBART 4** *(Acordo):* Nunca dois consumidores não Bizantinos entregam valores diferentes.
- **NBART 5** *(Evidência):* O observador confiável *TO* invoca *certify*(*TO*, *evidence*), ao fim de um intervalo finito de tempo após o início da transferência.
- **NBART 6** *(Certificação dos Produtores): hasProduced (evidence, p_i)* retorna *verdadeiro* se o produtor p_i participou, correctamente, na transferência do valor.
- **NBART 7** *(Certificação dos Consumidores): hasAcknowledged (evidence, c_j)* retorna *verdadeiro* se o consumidor c_j participou, correctamente, na transferência do valor.

Posto isto, é possível definir o objectivo de cada produtor p_i não Bizantino, que consiste em a invocação de *hasProduced (evidence, p_i)* retornar *verdadeiro*, enquanto cada consumidor não Bizantino c_j visa obter o valor correcto e garantir que *hasAcknowledged (evidence, c_j)* retorne *verdadeiro*.

4 Protocolo NBART

Seguidamente, apresenta-se o protocolo que visa resolver o problema NBART. Este é composto pelos Algoritmos 1, 2, e 3, executados pelos produtores, consumidores

Algorithm 1: Algoritmo NBART (produtor p_i)

```
01 upon init do
02    myvalue := myhash := myhashsig := ⊥;
05    round := 0;

06 upon produce() ∧ round = 0 do
07    produce(p_i,myvalue);
08    myhash := hash(myvalue);
09    myhashsig := sign (p_i, myhash);

10 upon nextRound ∧ round = 0 do // início da ronda 1
11    round := 1;
12    msgsig := sign (p_i, VALUE || myvalue || myhash || myhashsig);
13    forall c_j ∈ consumerset_i do
14       send (p_i, c_j, [VALUE, myvalue, myhash, myhashsig, msgsig])
15    msgsig := sign (p_i, SUMMARY || myhash || myhashsig);
16    forall c_j ∈ C\consumerset_i do
17       send (p_i, c_j, [SUMMARY, myhash, myhashsig, msgsig])
```

e pelo observador confiável, respectivamente. Para que o protocolo esteja correcto na presença de, no máximo, f produtores e f consumidores Bizantinos, requere-se que $N \geq 2f + 1$. De notar que $\mathcal{P}$, $\mathcal{C}$ e $\mathcal{F}$ representam os conjuntos de produtores, consumidores e nós Bizantinos, respectivamente ($|\mathcal{P}| = |\mathcal{C}| = N$).

4.1 Estrutura de Rondas

Modela-se o tempo de execução do algoritmo em rondas. Inicialmente, todos os participantes encontram-se na ronda 0. O evento *nextRound(n)* é activado sempre que se transita de ronda. Dado que se assume que o sistema é síncrono, definem-se as rondas de tal forma que, se um nó envia uma mensagem em resposta a um evento *nextRound*, essa mensagem é recebida até ao início da ronda seguinte.

O protocolo visa garantir que cada consumidor recebe informação suficiente que lhe permita determinar qual dos vários valores enviados por produtores é correcto. Com esse intuito, requere-se a cada produtor que envie uma cópia do valor a todos os consumidores. No entanto, dadas as propriedades de não colisão das funções *hash*, é suficiente que o produtor envie apenas a *hash* do valor a todos os consumidores. Desta forma, cada um dos consumidores é capaz de determinar qual a *hash* do valor correcto, baseando-se numa maioria ($f+1$) de *hashes* idênticas. Cada produtor tem, ainda, de enviar o valor a alguns consumidores, de forma a garantir que cada um deles consegue obter uma cópia do valor correcto. Mais propriamente, o valor tem de ser enviado pelo produtor p_i a todos os consumidores do conjunto determinado pela função determinista $consumerset_i$, cuja definição é a seguinte:

$$consumerset_i = \{c_j | j \in [i...(i+f) \ mod \ N]\}$$

A intuição desta formulação é a de que, a cada nó, é atribuído um índice, sendo estes índices vistos como um espaço circular, em que cada produtor é responsável por enviar o valor aos $f+1$ consumidores consecutivos que lhe seguem. Por exemplo, num sistema onde $f = 1$ e $N = 3$, $\mathcal{P} = \{p_1, p_2, p_3\}$, $\mathcal{C} = \{c_1, c_2, c_3\}$, $consumerset_1 = \{c_1, c_2, c_3\}$, $consumerset_2 = \{c_2, c_3, c_1\}$ e $consumerset_1 = \{c_3, c_1, c_2\}$.

O algoritmo tem uma duração de três rondas. Na primeira ronda, os produtores enviam os valores e as *hashes* aos consumidores, conforme especificado em cima. Na segunda ronda, os consumidores enviam informação assinada ao observador

Algorithm 2: Algoritmo NBART (consumidor c_j)

```
01 upon init do
02    myvalue := myhash:= ⊥;
04    confirm := values := [⊥]^P;
06    round := 0;

07 upon nextRound ∧ round = 0 do // start of round 1
08    round := 1;

09 upon deliver (p_i, c_j, [VALUE, pvalue, phash, phashsig, msgsig]) ∧ round = 1 do
10    if (c_j ∈ consumerset_i)then
11       if verifysig(p_i, VALUE || pvalue || phash || phashsig, msgsig)then
12          if verifysig(p_i,phash, phashsig) then
13             if verifyhash(pvalue, phash) then
14                values[p_i] := ⟨pvalue, phash, phashsig⟩;

15 upon deliver (p_i, c_j, [SUMMARY, phash, phashsig, msgsig]) ∧ round = 1 do
16    if (c_j ∉ consumerset_i)then
17       if verifysig(p_i, SUMMARY ||phash || phashsig, msgsig) then
18          if verifysig(p_i,phash, phashsig) then
19             values[p_i] := ⟨⊥, phash, phashsig⟩;

20 upon nextRound ∧ round = 1 do // start of round 2
21    round := 2;
22    myhash := h : #({p|value[p] = ⟨∗, h, ∗⟩}) > f.
23    myvalue := v : {p|value[p] = ⟨v, myhash, ∗⟩}.
24    forall p_i: values[p_i] = ⟨*, myhash, *⟩ do
25       confirm[p_i] := ⟨values[p_i].hash, values[p_i].signature⟩;
26    confsig := sign (c_j, confirm);
27    msgsig := sign (c_j, CERTIFICATE||confirm||confsig);
28    send (c_j, TO, [CERTIFICATE, confirm, confsig, msgsig])
29    consume (c_j, myvalue);
```

confiável. Finalmente, o observador efectua a certificação de todos os participantes durante a terceira ronda.

4.2 Operação dos Participantes

Elaboramos agora uma descrição mais detalhada da operação de cada participante. Inicialmente, cada produtor p_i produz o valor, gera a sua *hash* e assina-a (Alg. 1, linhas 7-9). Durante a ronda 1, o valor, conjuntamente com a *hash* e respectiva assinatura, são enviados para todos os consumidores de $consumerset_i$ (Alg. 1, linhas 12-14), enquanto apenas a *hash* e a sua assinatura são enviadas para os restantes consumidores (Alg. 1, linhas 15-17).

Por sua vez, cada consumidor espera, durante a ronda 1, por todas as mensagens contendo o valor ou, apenas, a sua *hash*, e guarda essa informação no vector *values* (Alg. 2, linhas 14 e 19). De realçar o facto de, caso algum produtor não envie a informação apropriada, a posição correspondente a esse produtor no vector *values* conterá, no final da ronda 1, o valor $\perp$. Isto servirá de testemunho contra o produtor, perante a avaliação do observador confiável.

Após o evento *nextRound* que dá início à ronda 2, cada consumidor adopta o valor v cuja *hash* aparece numa maioria ($f+1$) de posições do vector *values* (Alg. 2, linhas 22-23). Tendo em conta o facto de haver no máximo f produtores Bizantinos, garante-se que existe apenas um valor que satisfaz esta condição de maioria. De seguida, cada consumidor gera um vector de confirmação assinado e envia-o ao observador confiável (Alg. 2, linhas 24-28), terminando a execução do protocolo com o

Algorithm 3: Algoritmo NBART (observador confiável TO)

```
01 upon init do
02    evidence:= [⊥]^C;
03    round := 0;

04 upon nextRound ∧ round < 2 do
05    round := round+1;

06 upon deliver (c_j, TO, [CERTIFICATE, confirm, confsig, msgsig]) ∧ round = 2 do
07    if verifysig (c_j, CERTIFICATE||confirm||confsig, msgsig) then
08       if verifysig (c_j, confirm, confsig) then
09          evidence[c_j] := ⟨confirm, confsig⟩;

10 upon nextRound ∧ round = 2 do // start of round 3
11    certify (TO, evidence);
```

consumo do valor correcto (Alg. 2, linha 29). O vector de confirmação terá de conter, para a posição correspondente a cada produtor p_i, a *hash* e a respectiva assinatura ou o valor $\perp$, quando nenhuma informação correctamente assinada foi enviada por p_i. Assim, o conjunto de todas as estruturas assinadas e enviadas por consumidores, reunidos durante a ronda 2 (Alg. 3, linhas 7-9), providencia informação suficiente para o observador confiável tomar a decisão acerca de quem deve ser recompensado (Alg. 3, linha 11).

4.3 Certificação dos Participantes

Considerando a estrutura de dados *evidence* criada pelo observador confiável, apresenta-se agora uma definição mais precisa das funções *hasProduced* e *hasAcknowledged*. Para o efeito, usa-se $h(v)$ para denotar a *hash* do valor v e $s_{p_k}(h(v))$ para denotar a assinatura de $h(v)$ por parte do produtor p_k.

Assim, *hasProduced (evidence, p_i)* retorna *verdadeiro* se existirem $N - f$ consumidores $c_k \in \{c_1, \ldots, c_{N-f}\}$ tais que $evidence[c_k][p_i] = \langle h(v), s_{p_i}(h(v))\rangle$. Caso contrário, retorna *falso*.

Já *hasAcknowledged (evidence, c_j)* retorna *verdadeiro* se existir um conjunto de produtores, denominado por $correctset_j$, tal que $|correctset_j| = N - f$ e $\forall p_k \in correctset_j$ *hasProduced*($evidence$, p_k) retorna *verdadeiro* e $evidence[c_j][p_k] = \langle h(v), s_{p_k}(h(v))\rangle$. Caso contrário, retorna *falso*.

5 Análise

A análise do protocolo proposto é composta por duas partes. Inicialmente, prova-se a sua correcção num ambiente apenas com participantes Bizantinos e Altruístas. Segue-se uma demonstração de que o protocolo é um equilíbrio de Nash, e que, portanto, é do interesse de qualquer participante Racional seguir o comportamento especificado pelo protocolo.

5.1 Correcção

A presente secção providencia uma prova da correcção do protocolo, de acordo com as propriedades NBART, assumindo que todos os participantes não Bizantinos respeitam o protocolo, ou seja, produzem o mesmo valor v e enviam-no a todos os consumidores de *consumerset*, enviando, também, a respectiva *hash*, juntamente

com a assinatura, para todos os consumidores. Adicionalmente, assume-se que, se $h(v') = h(v)$, para alguma função *hash* h, então $v = v'$.

Começa-se por realçar a seguinte propriedade, que garante que cada consumidor está associado a $f + 1$ produtores responsáveis por enviar-lhe o valor.

Propriedade 1. Seja $producerset_j$ o conjunto de produtores que têm a obrigação de enviar o valor ao consumidor c_j. Então, $\forall_{c_j \in C}, |producerset_j| = f + 1$.

Através desta propriedade, é possível provar a correcção do protocolo nos seguintes Lemas.

Lema 1. (Validade) *Se um consumidor não Bizantino entrega v', então v' foi proposto por algum produtor não Bizantino e é igual ao valor correcto v.*

Demonstração. Um consumidor não Bizantino c só entrega v' se tiver recebido $h(v')$ de, pelo menos, $f + 1$ produtores e o próprio valor v' de, pelo menos, um produtor. Dado que existem, no máximo, f produtores Bizantinos, então c, necessariamente, recebeu $h(v')$ de, pelo menos, um produtor não Bizantino. Pela hipótese de que todos os produtores não Bizantinos enviam o valor correcto, é verdade que $h(v') = h(v)$, o que implica que $v = v'$, e, consequentemente, v' terá sido enviado por algum produtor não Bizantino.

Lema 2. (Integridade) *Nenhum consumidor não Bizantino invoca a primitiva* consume *mais do que uma vez.*

Demonstração. Por contradição, assuma-se que é possível que um consumidor não Bizantino c_j invoque a primitiva *consume* mais do que uma vez, para entregar os valores $v_1, v_2, ...v_n$, para $n \geq 2$. Um consumidor Bizantino só entrega v_i se tiver recebido $f + 1$ ou mais assinaturas da *hash* desse valor, bem como o próprio valor. Pelo Lema 1, existem necessariamente n produtores não Bizantinos $p_1, p_2, \ldots p_n$ que propuseram os valores entregues por c_j. Se $v_1 = v_2 = ... = v_n = v$, então o consumidor estaria a violar o protocolo ao entregar o mesmo valor mais do que uma vez. Se, pelo contrário, o consumidor entregar v_i e v_j tal que $v_i \neq v_j$, então terão de existir dois produtores p_i e p_j não Bizantinos que enviaram v_i e v_j, respectivamente, o que contradiz a hipótese de que todos os produtores não Bizantinos produzem e enviam o mesmo valor. Portanto, não é possível que um consumidor não Bizantino invoque a primitiva *consume* mais do que uma vez.

Lema 3. (Acordo) *Nunca dois consumidores não Bizantinos entregam valores diferentes.*

Demonstração. Pelo Lema 1, se um consumidor não Bizantino entrega um valor v, então esse valor foi enviado por algum produtor não Bizantino. Por hipótese, como todos os produtores enviam o mesmo valor, nenhum outro consumidor pode conseguir reunir $f + 1$ *hashes* de algum valor v' diferente de v. Logo, nenhum outro consumidor entrega um valor diferente de v.

Lema 4. (Terminação) *Alguma vez, todos os consumidores não Bizantinos entregam o mesmo valor v.*

Demonstração. Por hipótese, todos os produtores não Bizantinos produzem e enviam o mesmo valor v ou a sua *hash* para todos os consumidores, no início da

ronda 1. Dado que os canais de comunicação são fiáveis e síncronos, e pela Propriedade 1, todos os consumidores não Bizantinos recebem $f + 1$ ou mais *hashes* de v bem como o próprio valor, antes do início da ronda 2. Quando o evento *nextRound* é activado para dar início à ronda 2, já todos os consumidores não Bizantinos possuirão a informação necessária que lhes permite entregar v até ao fim dessa ronda.

Lema 5. (Evidência) *O observador confiável invoca* certify(*TO, evidence*) *no início da ronda 3.*

Demonstração. Resulta, directamente, da construção do algoritmo.

Lema 6. (Certificação dos Produtores) *hasProduced (evidence, p_i) retorna* verdadeiro *se p_i participou, correctamente, na transferência do valor.*

Demonstração. Pelo Lema 4, todos os consumidores não Bizantinos entregam o mesmo valor v. O algoritmo dita que, antes de entregarem o valor, os consumidores enviam os vectores *confirm* ao observador confiável. Assumindo um máximo de f consumidores Bizantinos, garante-se que há $N - f$ consumidores não Bizantinos a enviar informação ao observador, no início da ronda 2. Se um produtor p_i obedeceu ao protocolo, cada um desses consumidores não Bizantinos terá recebido $h(v)$ de p_i e incluído $\langle h(v), s_{p_i}(h(v))\rangle$ na informação enviada ao observador. Como estes dados são incluídos na estrutura de dados *evidence*, então *hasProduced (evidence,p_i)* retorna *verdadeiro.*

Lema 7. (Certificação dos Consumidores) *hasAcknowledged (evidence, c_j) retorna verdadeiro se c_j participou, correctamente, na transferência do valor.*

Demonstração. Por construção, um consumidor não Bizantino envia o seu vector *confirm* ao observador confiável. Para além disso, cada consumidor c_j inclui $\langle h(v), s_{p_i}(h(v))\rangle$ nesse vector, para cada produtor p_i que lhe enviou a informação correcta. De acordo com o Lema 6 e dado que existem, no máximo, f produtores Bizantinos, garante-se que o vector *confirm* de c_j contém a informação correcta enviada por, no mínimo, $N - f$ produtores, e garante-se que, para cada um desses produtores p_k, *hasProduced(evidence,p_k)* retorna *verdadeiro.* Por isso, *hasAcknowledged (evidence, c_j)* retorna *verdadeiro.*

Teorema 1. (Correcção) *Se todos os participantes não Bizantinos obedecerem ao protocolo e se houver, no máximo f produtores e f consumidores Bizantinos, então o protocolo proposto resolve o problema NBART.*

Demonstração. Resulta, directamente, dos Lemas 1, 2, 3, 4, 5, 6, e 7.

5.2 Análise Racional

De forma a provar que o protocolo é um equilíbrio de Nash, modela-se o problema NBART como um jogo estratégico $\Gamma = (M, S_M, \boldsymbol{u})$ em que M representa o conjunto de todos os jogadores, tal que $M = \mathcal{P} \cup \mathcal{C}$, sendo $\mathcal{P}$ o conjunto de produtores e $\mathcal{C}$ o conjunto de consumidores. Note-se que estes conjuntos são disjuntos, *i.e*, $\mathcal{P} \cap \mathcal{C} = \emptyset$. Cada jogador i adopta um plano de acção do conjunto S_i (estratégia), uma única vez, válido para todas as suas jogadas futuras. Estas decisões são feitas simultaneamente, e, como os jogadores Racionais não estão em conluio, são feitas de

forma independente. Define-se, como perfil de estratégias, a correspondência entre jogadores e a sua estratégia: $\boldsymbol{\sigma}_M : M \mapsto S_M$. σ_i denota a estratégia adoptada por i.

A função utilidade de cada jogador i é representada por $u_i(\boldsymbol{\sigma}) = \mathit{benefits}_i(\boldsymbol{\sigma}) - \mathit{costs}_i(\boldsymbol{\sigma})$, em que *benefits* determina os benefícios e *costs* representa os custos, quando todos os jogadores seguem a estratégia especificada por $\boldsymbol{\sigma}$. Assume-se que os produtores e consumidores obtêm, respectivamente, os benefícios β_P e β_C se atingirem os seus objectivos, que se assume serem superiores a quaisquer custos. Representa-se por $\mathcal{F} = \mathcal{F}_\mathcal{P} \cup \mathcal{F}_\mathcal{C}$ o conjunto de todos os produtores Bizantinos ($\mathcal{F}_\mathcal{P}$) e de consumidores Bizantinos ($\mathcal{F}_\mathcal{C}$). Não se assume qualquer restrição em relação ao comportamento Bizantino, exceptuando a hipótese de que $|\mathcal{F}_\mathcal{P}| \leq f$ e $|\mathcal{F}_\mathcal{C}| \leq f$. Por fim, denota-se por $\boldsymbol{\pi}_P \in \mathit{\Pi}_P$ e $\boldsymbol{\pi}_C \in \mathit{\Pi}_C$ o perfil de estratégias de todos os produtores e consumidores Bizantinos, respectivamente.

Tal como no modelo definido em [7], assume-se que os jogadores são avessos ao risco. Isto significa que a função utilidade de cada jogador i contempla o pior cenário possível de comportamento Bizantino, ou seja:

$$u_i(\boldsymbol{\sigma}_M) = \min_{x,y \in [0 \ldots f]} \circ \min_{F_P : |F_P| = x, F_C : |F_C| = y} \circ \min_{\boldsymbol{\pi}_P \in \mathit{\Pi}_P, \boldsymbol{\pi}_C \in \mathit{\Pi}_C} u_i(\boldsymbol{\sigma}_{M-F}, \boldsymbol{\pi}_P, \boldsymbol{\pi}_C) \quad (1)$$

Assim, define-se um equilíbrio de Nash como sendo um perfil de estratégias para o qual nenhum jogador possui qualquer incentivo em desviar-se da estratégia especificada por esse perfil. Isto é:

Definicão 1. *$\boldsymbol{\sigma}$ é um equilíbrio de Nash se* $\forall_{i \in M-F} \forall_{\sigma_i^* \in S_i} u_i(\boldsymbol{\sigma}) \geq u_i(\boldsymbol{\sigma}_{M-i}, \sigma_i^*)$.

Nos seguintes Lemas, prova-se que nenhum produtor ou consumidor beneficia de desviar-se do protocolo.

Lema 8. *Para cada produtor $p \in \mathcal{P}$, seja $\boldsymbol{\sigma}$ um perfil alternativo de estratégias para p, em que p envia a* hash *do valor e respectiva assinatura a todos os consumidores de st_p, para qualquer conjunto $st_p \subset \mathcal{C}$ tal que $|st_p| = k$ e $k < N$. Assim,* $\text{benefits}_p(\boldsymbol{\sigma}^*) = 0$.

Demonstração. Os jogadores Racionais determinam a sua utilidade considerando o pior caso de comportamento Bizantino (Equação 1). Suponha-se que $F_C \subseteq st_p$, para $|\mathcal{F}_\mathcal{C}| = f$. Neste caso, se p apenas envia o valor ou a sua *hash* a $k < N$ consumidores e se nenhum consumidor Bizantino envia a sua confirmação ao observador confiável *TO*, então *TO* apenas recebe $k - f < N - f$ confirmações com o valor $\langle h(v), s_p(h(v)) \rangle$, durante a segunda ronda. Por isso, *hasProduced (evidence, p)* retornará *falso*, e, consequentemente, $\mathit{benefits}_p(\boldsymbol{\sigma}^*) = 0$.

Lema 9. *Para cada produtor $p_i \in \mathcal{P}$, seja $\boldsymbol{\sigma}^*$ um perfil alternativo de estratégias, em que p_i não envia o valor a todos os consumidores de* consumerset_i. *Assim,* $\text{benefits}_{p_i}(\boldsymbol{\sigma}^*) = 0$.

Demonstração. Para que *hasProduced (evidence, p_i)* retorne *verdadeiro*, é necessário que o observador confiável receba a informação $\langle h(v), s_{p_i}(h(v)) \rangle$ de $N - f$ consumidores. Como o conjunto $\mathit{consumerset}_i$ contém $f + 1$ elementos, o observador tem de receber essa informação de, pelo menos, um elemento desse conjunto. No pior caso, $|\mathcal{F}_\mathcal{C}| = f$ e $\mathcal{F}_\mathcal{C} \subseteq \mathit{consumerset}_i$, e todos os consumidores de $\mathit{consumerset}_i$ a quem p_i envia o valor são Bizantinos e não enviam o certificado ao observador. Nesta situação, *hasProduced (evidence, p_i)* retornará *falso*, e, consequentemente, $\mathit{benefits}_{p_i}(\boldsymbol{\sigma}^*) = 0$.

Teorema 2. (Equilíbrio de Nash para Produtores) *Nenhum produtor tem qualquer incentivo em desviar-se, unilateralmente, do protocolo.*

Demonstração. Seja $\boldsymbol{\sigma}$ o perfil de estratégias em que cada participante não Bizantino obedece ao protocolo. Dos Lemas 8 e 9 resulta que, se um produtor p segue uma estratégia alternativa $\boldsymbol{\sigma}^*$, então não obtém benefícios, no pior caso, e $u_p(\boldsymbol{\sigma}^*) = -costs_p < 0$. De acordo com o Teorema 1, $u_p(\boldsymbol{\sigma}) = \beta_P - costs_p(\boldsymbol{\sigma}) > 0$ (β_P é superior a quaisquer custos). Portanto, $u_p(\boldsymbol{\sigma}) > u_p(\boldsymbol{\sigma}^*)$.

Lema 10. *Para cada consumidor $c \in \mathcal{C}$, seja $\boldsymbol{\sigma}^*$ um perfil de estratégias alternativo, em que c não envia o vector confirm para o observador confiável. Então,* $\text{benefits}(\boldsymbol{\sigma}^*) = 0$.

Demonstração. Deriva, directamente, da definição de *hasAcknowledged*.

Lema 11. *Para cada consumidor $c \in \mathcal{C}$, seja $receive_c$ o conjunto de produtores que enviou o valor correcto ou a* hash *desse valor ao consumidor c, onde $K = |receive_c|$. Denote-se por $\boldsymbol{\sigma}^*$ um perfil de estratégias alternativo, em que c envia um vector* confirm *incompleto ao observador confiável, com, apenas, k entradas ($f + 1 \leq k < K$) que não contêm o valor $\perp$. Assim,* $\text{benefits}_c(\boldsymbol{\sigma}^*) = 0$.

Demonstração. O pior caso possível para um consumidor não Bizantino c_j occorre quando $|F_P| = f$ e todos os produtores Bizantinos enviam informação válida a c_j, não enviando essa informação aos restantes consumidores. Neste caso, apenas existe um conjunto $correctset_j$ que só contém produtores não Bizantinos. Se c_j não inclui o valor $\langle hash(v), s_{p_i}(hash(v))$ no seu vector e se p_i não é Bizantino, então a estrutura de dados *evidence* não conterá, para o consumidor c_j, um certificado de todos os produtores de $correctset_j$. Assim, *hasAcknowledged(evidence,c)* retorna *falso* e $\text{benefits}_c(\boldsymbol{\sigma}^*) = 0$.

Teorema 3. (Equilíbrio de Nash para Consumidores) *Nenhum consumidor possui incentivos para se desviar, unilateralmente, do protocolo.*

Demonstração. Seja $\boldsymbol{\sigma}$ o perfil de estratégias em que cada participante não Bizantino obedece ao protocolo. Pelos Lemas 10 e 11, $u_c(\boldsymbol{\sigma}^*) = -costs(\boldsymbol{\sigma}^*) < 0$. De acordo com o Teorema 1, $u_c(\boldsymbol{\sigma}) = \beta_C - costs_c(\boldsymbol{\sigma}) > 0$, ($\beta_C$ é superior a quaisquer custos). Portanto, $u_c(\boldsymbol{\sigma}) > u_c(\boldsymbol{\sigma}^*)$.

6 Conclusões

Este artigo introduziu o problema NBART, onde se visa garantir a transferência fiável de dados de um conjunto de produtores para um conjunto de consumidores, em que o comportamento dos participantes é caracterizado pelo modelo BAR. Este é um problema de relevo e que facilita a construção de um sistema de computação voluntária, baseado numa arquitectura entre pares, e capaz de resolver problemas mais complexos do que os resolvidos pelo sistema Boinc[2], pelo facto de permitir a comunicação directa entre tarefas.

Propôs-se um protocolo que resolve o problema em questão para $N \geq 2f + 1$, em que N corresponde à cardinalidade de ambos os conjuntos de produtores e de consumidores. Através de uma análise baseada nos conceitos da Teoria de Jogos[16], demonstrou-se que o protocolo proposto é um Equilíbrio de Nash, o que

significa que nenhum participante Racional possui qualquer incentivo em não seguir o comportamento especificado pela solução proposta.

Como trabalho futuro, visa-se desenvolver um sistema de armazenamento entre pares, que tire proveito dos recursos disponibilizados por voluntários para suportar computação distribuída. O principal objectivo é usar o modelo *map-reduce* para executar tarefas de forma paralela. Neste cenário, o algoritmo NBART torna-se um componente importante para a transferência de dados entre *mappers* e *reducers* e para preservar a informação ao longo do tempo, através de sucessivas operações de transferência.

Agradecimentos

Este trabalho foi parcialmente suportado pela FCT através do projecto "HPCI" (PTDC/EIA-EIA/102212/2008) e através do financiamento multianual do INESC-ID com fundos do Programa PIDDAC.

Referências

1. Aiyer, S., Alvisi, L., Clement, A., Dahlin, M., Martin, J.P., Porth, C.: BAR fault tolerance for cooperative services. In: SOSP'05. pp. 45–58. Brighton, United Kingdom (Oct 2005)
2. Anderson, D.: Boinc: A system for public-resource computing and storage. In: GRID'04. pp. 4–10. Pittsburgh, USA (Nov 2004)
3. Anderson, D., Cobb, J., Korpela, E., Lebofsky, M., Werthimer, D.: SETI@home: an experiment in public-resource computing. Communications of the ACM 45(11), 56–61 (November 2002)
4. Axelrod, R.: The Evolution of Cooperation. Basic Books, New York (1984)
5. Bracha, G., Toueg, S.: Asynchronous consensus and broadcast protocols. J. ACM 32, 824–840 (October 1985)
6. Castro, M., Liskov, B.: Practical byzantine fault tolerance and proactive recovery. ACM Transactions on Computer Systems 20(4), 398–461 (2002)
7. Clement, A., Li, H., Napper, J., Martin, J.P., Alvisi, L., Dahlin, M.: BAR primer. In: DSN'08. pp. 287–296. Anchorage, Alaska, USA (Jun 2008)
8. Cohen, B.: Incentives build robustness in bittorrent. In: NetEcon'03. Berkeley (CA), USA (Jun 2003)
9. Dolev, D., Strong, H.: Authenticated algorithms for byzantine agreement. SIAM J. Comput. 12(4), 656–666 (1983)
10. Hardin, G.: The tragedy of the commons. Science 162(3859), 1243–47 (1968)
11. Lamport, L., Shostak, R., Pease, M.: The byzantine generals problem. ACM Trans. Program. Lang. Syst. 4, 382–401 (July 1982)
12. Li, H., Clement, A., Marchetti, M., Kapritsos, M., Robison, L., Alvisi, L., Dahlin, M.: Flightpath: Obedience vs choice in cooperative services. In: OSDI'08. San Diego, CA, USA (Dec 2008)
13. Li, H., Clement, A., Wong, E., Napper, J., Roy, I., Alvisi, L., Dahlin, M.: BAR gossip. In: OSDI'06. pp. 191–204. Seattle, WA, USA (Nov 2006)
14. Malkhi, D., Reiter, M.: Byzantine quorum systems. In: STOC'97. pp. 569–578. El Paso, USA (1997)
15. Martin, J.P., Alvisi, L., Dahlin, M.: Minimal byzantine storage. In: DISC'02. pp. 311–325. Toulouse, France (Oct 2002)
16. Martin, O., Ariel, R.: A Course in Game Theory. MIT Press (1994)
17. Mokhtar, S., Pace, A., Quéandma, V.: FireSpam: Spam resilient gossiping in the BAR model. In: SRDS'10. pp. 225–234. New Delhi, India (Oct 2010)
18. Shneidman, J., Parkes, D., Massoulie, L.: Faithfulness in internet algorithms. In: PINS'2004. Portland, OR, USA (September 2004)