Cálculo Distribuido de Permutaciones y sus Aplicaciones al Juego Electrónico

Jordi Castellà-Roca, Vanesa Daza,
Josep Domingo-Ferrer y Francesc Sebé

Dept. d'Enginyeria Informàtica i Matemàtiques, Universitat Rovira i Virgili,
Av. Països Catalans 26, E-43007 Tarragona

{jordi.castella, vanesa.daza, josep.domingo, francesc.sebe}@urv.cat

Resumen Con la introducción de las nuevas tecnologías y, más en particular, la mejora de las comunicaciones, en la última década se ha incrementado considerablemente el número de operaciones que se realizan a través de una red de comunicaciones tal como Internet. Los protocolos de computación multiparte representan una herramienta especialmente útil en estos entornos ya que permiten calcular de manera distribuida y segura una determinada función entre un conjunto de usuarios conectados a través de la red. En este artículo nos centramos en el problema concreto de la generación de una permutación aleatoria de forma conjunta entre unos participantes de manera que la permutación resultante quede distribuida mediante un sistema de compartición de secretos. Finalmente, justificamos como este protocolo resulta especialmente útil para el juego electrónico.

1. Introducción

El problema de la *computación multiparte segura* fue introducido por Yao [19]. Más concretamente, Yao introdujo el conocido *problema del millonario*: dos millonarios quieren saber cual de los dos es más rico sin revelar ninguna información de cual es su riqueza. En general, la computación multiparte segura se ocupa de, dado un conjunto de participantes $P = \{P_1, \ldots, P_n\}$ que tienen cierta información secreta $s_1, \ldots, s_n$ respectivamente, cómo pueden calcular el resultado de una determinada función f de n variables en el punto $(s_1, \ldots, s_n)$ sin revelar su información secreta.

Los protocolos de computación multiparte permiten modelar un gran número de situaciones cotidianas como puede ser una votación electrónica o jugar a cartas a través de Internet, proporcionando así una solución a problemas asociados a dichas situaciones. En la literatura se han tenido en cuenta numerosos escenarios con el fin de garantizar la seguridad en todos ellos.

Las primeras propuestas que permitían calcular cualquier función de manera distribuida entre un conjunto de participantes se encuentran en [6,15]. Desde entonces, la literatura en el tema ha crecido considerablemente (por ejemplo,

[8,9,14]), considerando diferentes escenarios posibles y proporcionando protocolos seguros para la computación multiparte en estos escenarios. No obstante gran parte de estas soluciones genéricas no resultan eficientes dado un problema concreto. Por ese motivo, recientemente han surgido propuestas de protocolos de computación multiparte que resuelven de manera eficiente el cálculo de determinadas funciones [1,12]. En este artículo nos centramos en el caso en el que las entradas $s_1, \ldots, s_n$ de los participantes son fragmentos de un cierto secreto s que ha sido distribuido mediante un esquema de compartición de secretos. En [8] se prueba cómo es posible reducirnos a este caso, sin perder la generalidad.

Tras describir algunos de los protocolos distribuidos que permiten calcular de manera eficiente algunas de las operaciones más básicas entre secretos distribuidos, proponemos un protocolo que permite a un conjunto de participantes calcular de manera distribuida una permutación aleatoria sin que ninguno de los participantes tenga ningún tipo de información sobre cuál es la permutación que se ha generado de manera conjunta.

Nos restringimos al caso en el que los participantes siguen el protocolo correctamente. En el caso en el que no sea cierta esta suposición es posible extender los protocolos que presentamos utilizando esquemas para compartir secretos verificables así como esquemas de compromiso.

El resto del artículo se estructura como sigue. En la Sección 2 describimos algunos protocolos que permiten efectuar algunas operaciones básicas entre secretos distribuidos y analizamos su coste considerando el número de mensajes intercambiado durante su ejecución. En la Sección 3 introducimos, basado en los protocolos descritos anteriormente, un protocolo que permite calcular de manera distribuida una permutación entre un conjunto de participantes. En la Sección 4 mostramos la aplicación de los protocolos anteriores al juego electrónico. Finalmente, concluimos en la Sección 5, planteando futuras líneas de investigación relacionadas.

2. Preliminares

Supongamos un conjunto de participantes $P = \{P_1, \ldots, P_n\}$ que tiene fragmentos de un cierto secreto. En esta sección revisamos algunos protocolos de computación multiparte segura que permiten calcular de manera eficiente determinadas funciones básicas de los secretos a partir de sus fragmentos. Una descripción de estos protocolos se puede encontrar en [1,7].

Dado el conjunto de participantes $P = \{P_1, \ldots, P_n\}$ y un secreto s, notaremos $[s]_\Sigma$ al conjunto de fragmentos resultante de aplicar un esquema para compartir secretos entre el conjunto de participantes P sobre el secreto s. Supondremos que el esquema para compartir secretos que se utiliza es el esquema de Shamir [18], aunque los protocolos que presentamos son extensibles a cualquier esquema para compartir secretos lineal. En este esquema el secreto s es distribuido entre el conjunto de participantes P a partir de un polinomio aleatorio de grado $t-1$ cuyo término independiente es el secreto s. El secreto s pertenece a un cuerpo finito K cuyo cardinal cumple que $|K| > n$. El fragmento de cada uno de los

participantes corresponde al polinomio evaluado en un punto conocido asociado a cada participante. Es posible reconstruir el secreto a partir de al menos t fragmentos utilizando, por ejemplo, la interpolación de Lagrange.

2.1. Operaciones básicas entre secretos

- **Creación y distribución de un secreto.** Un participante de P que desea compartir un secreto s, genera el conjunto de n fragmentos $[s]_\Sigma$, se queda el fragmento que le corresponde y envía cada uno de los $(n-1)$ restantes al participante al que le corresponde. Durante esta operación se envía $(n-1)$ mensajes.
- **Suma de secretos.** Supongamos dos secretos s_1 y s_2 y que cada participante de P dispone de su fragmento correspondiente a $[s_1]_\Sigma$ y $[s_2]_\Sigma$. Dadas las propiedades lineales del esquema, los participantes pueden calcular individualmente sus fragmentos de la suma s_1+s_2 mediante la suma del fragmento de s_1 y del fragmento de s_2 que cada uno tiene. Notaremos esta operación como $[s_1 + s_2]_\Sigma = [s_1]_\Sigma + [s_2]_\Sigma$. Nótese que el protocolo es fácilmente extensible a la suma de más valores. Como inicialmente cada participante ya dispone de sus fragmentos, esta operación no implica comunicación.
- **Producto de un secreto por una constante.** Supongamos un secreto s y que cada participante de P dispone de su fragmento correspondiente de $[s]_\Sigma$. Dado un elemento $a \in K$ conocido, los participantes puede obtener su fragmento de $a \cdot s$ multiplicando por a el fragmento que ellos tienen del secreto s obteniendo así el fragmento de $[a \cdot s]_\Sigma$. Esta operación no requiere comunicación entre los participantes.
- **Suma de un secreto y una constante.** Supongamos un secreto s y que cada participante de P dispone de su fragmento correspondiente a $[s]_\Sigma$. Dado un elemento $a \in K$ conocido, vamos a describir como los participantes pueden obtener su fragmento de $a + s$.
  Un participante de P genera un conjunto de n fragmentos correspondiente a a (al cual notaremos $[a]_\Sigma$) y envía a cada participante el fragmento que le corresponde. Luego, los participantes calculan conjuntamente fragmentos de $a + s$ calculando $[a + s]_\Sigma = [a]_\Sigma + [s]_\Sigma$ utilizando el protocolo de suma de secretos.
  Nótese que esta operación es muy parecida a la suma de dos secretos. La diferencia yace en que en esta operación uno de los dos sumandos es conocido. En esta operación, uno de los participantes envía $(n-1)$ mensajes. Por tanto, el número total de mensajes es $(n-1)$.
- **Producto de secretos.** Supongamos dos secretos s_1 y s_2 y que cada participante de P dispone de su fragmento correspondiente a $[s_1]_\Sigma$ y $[s_2]_\Sigma$. Los participantes pueden calcular fragmentos del producto $s_1 \cdot s_2$ a partir de los fragmentos de los secretos s_1 y s_2 [8,14,15]. La solución a este problema es ligeramente más complicada que en el caso anterior de la suma. En efecto, si se intenta aplicar el mismo método y que cada participante calcule el producto de los fragmentos que tiene de s_1 y de s_2, el resultado es un fragmento del producto de los fragmentos, pero los parámetros del esquema para

compartir secretos varían, ya que son fragmentos de un cierto polinomio de grado $2(t-1)$. Además dicho polinomio no es totalmente aleatorio, ya que se trata de un polinomio que es producto de dos polinomios de grado $(t-1)$. Para solucionar estos problemas, cada participante reparte el producto de sus fragmentos utilizando el esquema para compartir secretos Σ. Una combinación adecuada de los fragmentos que cada participante recibe les permite calcular su fragmento adecuado del producto.
El número de mensajes necesario para multiplicar dos secretos es $n(n-1)$. Para multiplicar m secretos mediante este sistema es necesario ejecutar el protocolo reiteradamente $(m-1)$ veces. Por tanto, el producto de m secretos requiere el envío de $n(n-1)(m-1)$ mensajes.

- **Generación conjunta de un número aleatorio.** La generación de fragmentos de un número aleatorio $r \in K$ se reduce a la generación de n números aleatorios: uno cada uno de los participantes. Esto es, el participante P_i elige aleatoriamente $r_i \in K$, genera $[r_i]_\Sigma$ y envía a cada participante el fragmento que le corresponde. A continuación los participantes calculan los fragmentos de la suma $r = r_1 + \ldots + r_n$ mediante la operación $[r]_\Sigma = [r_1]_\Sigma + \ldots + [r_n]_\Sigma$. Esta operación también es conocida como reparto democrático de secretos. El número de mensajes enviados es $n(n-1)$.
- **Generación aleatoria de valores invertibles.** El objetivo de este protocolo es generar fragmentos de un elemento invertible $\rho \in K$, es decir, $[\rho]_\Sigma$ tal que $\rho \neq 0$. Para ello, los participantes generan de forma conjunta los fragmentos de dos números aleatorios r_1 y r_2 aplicando el protocolo anterior (enviando $2n(n-1)$ mensajes) y obtienen $[r_1]_\Sigma$ y $[r_2]_\Sigma$. Luego calculan $[r_1 \cdot r_2]_\Sigma$ (enviando $n(n-1)$ mensajes) y a continuación reconstruyen el secreto $r_1 \cdot r_2$ (suponiendo que cada participante envía su fragmento al resto de participantes, esto implica $n(n-1)$ mensajes). Si el resultado que se obtiene es diferente de cero, los participantes eligen uno de los dos valores generados aleatoriamente, por ejemplo r_1, como el valor aleatorio invertible que se buscaba. En el caso de que el resultado sea cero, los participantes repiten de nuevo el proceso.
Si el cardinal de K es suficientemente grande, la probabilidad de que $r_1 \cdot r_2 = 0$ es despreciable. En este caso, el número total de mensajes es $4n(n-1)$.
- **Cálculo de elementos inversos.** Supongamos un secreto s y que cada participante de P dispone de su fragmento correspondiente de $[s]_\Sigma$. El objetivo es que cada participante disponga de su fragmento de $[s^{-1}]_\Sigma$. Para ello, los participantes generan de manera distribuida los fragmentos de un cierto valor aleatorio invertible, es decir, calculan $[\rho]_\Sigma$ (enviando $4n(n-1)$ mensajes durante el proceso). A continuación calculan $[s \cdot \rho]_\Sigma$ a partir de $[s]_\Sigma$ y $[\rho]_\Sigma$ (con $n(n-1)$ mensajes) y reconstruyen el secreto $\alpha = s \cdot \rho$ (también con $n(n-1)$ mensajes). Finalmente los fragmentos del inverso se calculan a partir del producto de la constante α^{-1} y los fragmentos de $[\rho]_\Sigma$, ya que $\alpha^{-1}[\rho]_\Sigma = s^{-1}\rho^{-1}[\rho]_\Sigma = [s^{-1} \cdot \rho^{-1} \cdot \rho]_\Sigma = [s^{-1}]_\Sigma$.
El número total de mensajes enviados es $6n(n-1)$.

Los diferentes protocolos que acabamos de describir se refieren a elementos de un cuerpo finito K. Sin embargo, se pueden extender fácilmente a vectores

y matrices. En este caso, se distribuye cada elemento del vector o de la matriz como un elemento del cuerpo finito K.

3. Generación Distribuida de Fragmentos de una Permutación Aleatoria

En esta sección describiremos un método para generar de manera distribuida fragmentos de una permutación aleatoria π de ℓ elementos entre un conjunto de participantes P. El objetivo es que el conjunto de participantes $P_1, \ldots, P_n$ calcule de manera conjunta, fragmentos de π. Manteniendo la notación seguida hasta el momento, denotaremos a un tal conjunto de fragmentos como $[\pi]_{\Sigma}$.

Para ello, utilizaremos la matriz permutación asociada a una permutación π. Dada una permutación de ℓ elementos $\pi : \{1, \ldots, \ell\} \rightarrow \{1, \ldots, \ell\}$, la matriz permutación de π se define como la matriz M_π que se obtiene al permutar las filas de la matriz identidad $\ell \times \ell$ según π. Así cada una de las filas y de las columnas de la matriz M_π contiene exactamente un 1 mientras que el resto de las posiciones son 0. Esta representación permite calcular la composición de permutaciones mediante el producto de sus matrices.

Dadas dos permutaciones π y σ de ℓ elementos, la composición de las dos permutaciones se puede calcular multiplicando las matrices permutación asociadas a dichas permutaciones. Esto es, $M_{\pi\circ\sigma} = M_\pi \cdot M_\sigma$. El conjunto de las matrices permutación de $\ell \times \ell$ elementos con la operación producto tiene estructura de grupo.

A continuación describimos cómo un participante puede repartir una matriz permutación entre un conjunto de participantes P. Después describiremos cómo se puede generar de manera distribuida una permutación aleatoria entre este conjunto de participantes.

3.1. Distribución de una matriz permutación

Supongamos que un participante de P quiere compartir una matriz permutación, M_π, con el resto de participantes.

Dicho participante genera la matriz M_π y comparte cada una de las entradas de la matriz utilizando el esquema de compartición de secretos Σ. Durante esta operación se generan y reparten ℓ^2 secretos mediante el envío de $\ell^2(n-1)$ mensajes.

3.2. Generación distribuida de una permutación aleatoria

El objetivo de este protocolo es que el conjunto de participantes P obtenga los fragmentos de una matriz de permutación aleatoria. Los participantes no deben conocer el valor de la matriz generada (hasta que deseen recuperarla mediante la unión de fragmentos). Para ello, cada uno de los participantes genera una matriz permutación y reparte sus elementos entre el conjunto de participantes P utilizando un esquema para compartir secretos Σ. A continuación,

los participantes calculan el producto de todas las matrices obteniendo así una permutación aleatoria. El protocolo es como sigue:

Generación aleatoria de matrices permutación Cada participante $P_i \in P$ genera, de forma secreta, una matriz permutación M_{π_i}.

Distribución de las matrices Cada participante calcula los fragmentos de la matriz M_{π_i} y los reparte entre el conjunto de participantes P. Se obtiene y distribuye así el conjunto de fragmentos $[M_{\pi_1}]_\Sigma, \ldots, [M_{\pi_n}]_\Sigma$. En esta etapa se envía un total de $\ell^2 n(n-1)$ mensajes.

Cálculo de fragmentos del producto de las matrices Los participantes de P calculan fragmentos del producto de las matrices $M_{\pi_1}, \ldots, M_{\pi_n}$. Esto es, $[M_{\pi_1} \cdot \ldots \cdot M_{\pi_n}]_\Sigma$. Obtienen de esta manera fragmentos de una matriz permutación M_π, donde $\pi = \pi_1 \circ \ldots \circ \pi_n$.

Dadas dos matrizes cuadradas A y B de dimensión ℓ, cada elemento de $A \cdot B$ se calcula como $(a \cdot b)_{i,j} = \sum_{k=1}^{\ell} a_{i,k} \cdot b_{k,j}$. El cálculo de cada elemento implica el cálculo de ℓ productos y $(\ell - 1)$ sumas. Si los elementos de A y B están distribuidos, el cálculo de cada elemento de $A \cdot B$ implica enviar $\ell n(n-1)$ mensajes. Como en total $A \cdot B$ tiene ℓ^2 elementos, durante su cálculo se envían $\ell^3 n(n-1)$ mensajes.

Dado que durante el cálculo de $[M_{\pi_1} \cdot \ldots \cdot M_{\pi_n}]_\Sigma$ se realiza un total de $(n-1)$ productos de matrices, durante esta etapa se envía un total de $\ell^3 n(n-1)^2$ mensajes.

El coste de este protocolo está dominado por la etapa de multiplicación de las matrices. El coste, en número de mensajes enviados de este sistema para n participantes y permutaciones de ℓ elementos es $O(\ell^3 n^3)$.

Durante la descripción de estos protocolos hemos supuesto un escenario donde el adversario es pasivo, esto es, obtiene la información de aquellos participantes que corrompe pero no tiene capacidad para alterar la información de estos participantes. Es posible utilizar las técnicas de verificación de los esquemas para compartir secretos así como los esquemas de compromiso para extender la seguridad de estos protocolos frente a adversarios activos.

4. Aplicaciones al Juego Electrónico

En los juegos de cartas una acción básica es la mezcla de la baraja de cartas. Uno de los jugadores es el encargado de girar las cartas boca abajo, de manera que no se pueda saber su valor, y mezclarlas. El resto de jugadores ven las acciones de este jugador de manera que impiden que pueda hacer una acción deshonesta.

En los juegos de cartas distribuidos a través de redes de comunicaciones (conocidos como esquemas de *mental poker*) los jugadores no pueden ejercer este control visual sobre el jugador que mezcla las cartas. Por este motivo se utilizan básicamente dos estratégias. En la primera, existe una tercera parte de confianza (TPC) que realiza las operaciones. En la mayoria de casinos on-line éste ejerce

de TPC y participa de forma activa en el juego. El problema reside en que el casino está en una posición privilegiada respecto a los jugadores.

En la segunda, son los jugadores quienes, de forma conjunta mediante un protocolo seguro de computación multiparte, realizan la mezcla. La utilización de un protocolo criptográfico de computación multiparte permite garantizar que ningún jugador dispondrá de ninguna ventaja con el resto sin necesidad de que exista una TPC.

Una baraja de cartas mezclada se puede representar como una permutación de l valores, por ejemplo en el caso del poker $l = 52$, [2,4,10,11,13]. Por consiguiente los jugadores pueden utilizar los protocolos descritos en la sección 3 para obtener una permutación de forma distribuida, es decir, una baraja de cartas mezclada de forma segura. Un jugador obtendría una carta recuperando (mediante la reconstrucción del secreto) los valores de una fila de la matriz de permutación. Para evitar confabulaciones de jugadores, es necesario que la reconstrucción de los secretos requiera la participación de todos los participantes.

5. Conclusiones

En este artículo hemos revisado algunos protocolos básicos que permiten realizar operaciones entre elementos que han sido distribuidos mediante un sistema de compartición de secretos. A partir de estos protocolos hemos presentado un sistema que permite generar una matriz permutación aleatoria distribuida entre un conjunto de participantes. Este sistema puede ser utilizado para generar barajas de cartas mezcladas en el juego electrónico remoto.

Nuestra investigación futura se centrará en encontrar construcciones que permitan obtener el mismo resultado mediante el envío de un número más reducido de mensajes.

Agradecimientos

Este trabajo ha sido parcialmente financiado por el Ministerio Español de Educación y Ciencia a través del proyecto SEG2004-04352-C04-01 “PROPRIETAS”.

Referencias

1. J. Algesheimer, J. Camenisch and V. Shoup, “Efficient computation modulo a shared secret with application to the generation of shared safe-prime products”, in *Advances in Cryptology - CRYPTO’2002*, LNCS, vol. 2442, pp. 417-432, 2002.
2. I. Barany and Z. Furedi, “Mental poker with three or more players”, in *Technical report, Mathematical Institute of the Hungarian Academy of Sciences*, 1983.
3. J. Bar-Ilan and D. Beaver, “Non-Cryptographic Fault-Tolerant Computing in a Constant Number of Rounds”, in *Procs. of the ACM Symposium on Principles of Distributed Computation*, pp. 201-209, 1989.

4. J. Castellà-Roca, J. Domingo-Ferrer, A. Riera and J. Borrell, "Practical mental poker without a TTP based on homomorphic encryption", in *Progress in Cryptology-Indocrypt'2003*, LNCS, vol. 2904, pp. 280-294, 2003.
5. D. Catalano, R. Cramer, I. Damgaard, G. Di Crescenzo, D. Pointcheval and T. Takagi, "Contemporary Cryptology", Ed. Birkhäuser, 2004.
6. D. Chaum, C. Crepeau and I. Damgaard, "Multiparty unconditionally secure protocols", in *Procs. of the twentieth annual ACM symposium on Theory of computing*, ACM Press, pp. 11-19, 1988.
7. R. Cramer and I. Damgaard, "Secure distributed linear algebra in a constant number of rounds", in *Advances in Cryptology - CRYPTO'2001*, LNCS, vol. 2139, pp. 119-136, 2001.
8. R. Cramer, I. Damgaard, and U. Maurer, "General secure multi-party computation from any linear secret-sharing scheme", in *Advances in Cryptology - EUROCRYPT'2000*, LNCS, vol. 1807, pp. 316-334, 2000.
9. R. Cramer, V. Daza, I. Gracia, J. Jimenez, G. Leander, J. Marti-Farre, C. Padro, "On Codes, Matroids and Secure Multi-Party Computation from Linear Secret Sharing Schemes", in *Advances in Cryptology - CRYPTO'2005*, LNCS, vol. 3621, pp. 327-343, 2005.
10. C. Crépeau, "A secure poker protocol that minimizes the effect of player coalitions", in, *Advances in Cryptology - CRYPTO'1985*, LNCS, vol. 218, pp. 73-86, 1985.
11. C. Crépeau, "A zero-knowledge poker protocol that achieves confidentiality of the players' strategy or how to achieve an electronic poker face", in *Advances in Cryptology - CRYPTO'1986*, LNCS, vol. 263, pp. 239-250, 1986.
12. I. Damgaard, M. Fitzi, E. Kiltz, J.B. Nielsen and T. Toft, "Unconditionally Secure Constant-Rounds Multi-Party Computation for Equality, Comparison, Bits and Exponentiation", in *Procs. of the third Theory of Cryptography Conference, TCC'2006*, LNCS, vol. 3876, pp. 285-304, 2006.
13. S. Fortune and M. Merritt, "Poker protocols", in *Advances in Cryptology - CRYPTO'1984*, LNCS, vol. 196, pp. 454-464, 1985.
14. R. Gennaro, M. O. Rabin and T. Rabin, "Simplified VSS and fast-track multiparty computations with applications to threshold cryptography", in *Procs. of 17th ACM Symposium Annual on Principles of Distributed Computing*, pp. 101–111, 1998.
15. O. Goldreich, S. Micali and A. Wigderson, "How to play any mental game", in *Procs. of the nineteenth annual ACM conference on Theory of computing*, ACM Press, pp. 218–229, 1987.
16. T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing", in *Advances in Cryptology – CRYPTO'91*, LNCS, vol. 576, pp. 129–140, 1991.
17. T. Rabin and M. Ben-Or, "Verifiable Secret Sharing and Multiparty Protocols with Honest Majority", in *Procs. of 21st STOC*, ACM Press, pp. 73–85, 1989.
18. A. Shamir, "How to share a secret", in *Communications of the ACM*, vol. 22, n.11, pp. 612-613, 1979.
19. A.C Yao, "Protocols for secure computation", in *Procs. of the 23rd Annual IEEE Symposium on Foundations of Computer Science*, pp. 160-164, 1982.