Groupes et cryptographie

Hamish Short,
Département de Mathématiques, Université de Aix–Marseille

Institut de Mathématiques de Marseille UMR 7353

21 février 2014

Groupes et cryptographie

Hamish Short,
Département de Mathématiques, Université de Aix–Marseille

Institut de Mathématiques de Marseille UMR 7353

21 février 2014

Cryprographie via secret partagé, Diffie Hellman.

Compliquée par deux collections d'auteurs (1999): Anshel, Anshel et Goldfeld, et de Ko, Lee, Cheon, Han, Kang et Park.

1. Nombre premier N et $0 < g < N$. Les entiers mod N = groupe cyclique additif.
2. Alice choisit un entier $0 < a < N$, Bob choisit $0 < b < N$.
3. Alice calcule $g^a = \overline{A} \mod N$. Bob calcule $g^b = \overline{B} \mod N$.
4. Alice et Bob échange les resultats $\overline{A}$ et $\overline{B}$.
5. Alice reçoit $\overline{B}$ et avec a, calcule $(\overline{B})^a = g^{(ab)} \mod N$.
6. Bob reçoit $\overline{A}$ et avec b, calcule $(\overline{A})^b = g^{(ba)} \mod N$.
7. Avec ce secret partagé, on peut envoyer des messages.

SECURITÉ = difficulté de **problème du logarithme discret** :

Données : $g, \overline{A}, N \in \mathbb{N}$, et $\exists a$, $0 < a < N$ t.q. $g^a = \overline{A} \mod N$.

Problème : retrouver a.

Cryprographie via secret partagé, Diffie Hellman.

Compliquée par deux collections d'auteurs (1999): Anshel, Anshel et Goldfeld, et de Ko, Lee, Cheon, Han, Kang et Park.

1. Nombre premier N et $0 < g < N$. Les entiers mod N = groupe cyclique additif.
2. Alice choisit un entier $0 < a < N$, Bob choisit $0 < b < N$.
3. Alice calcule $g^a = \overline{A} \mod N$. Bob calcule $g^b = \overline{B} \mod N$.
4. Alice et Bob échange les resultats $\overline{A}$ et $\overline{B}$.
5. Alice reçoit $\overline{B}$ et avec a, calcule $(\overline{B})^a = g^{(ab)} \mod N$.
6. Bob reçoit $\overline{A}$ et avec b, calcule $(\overline{A})^b = g^{(ba)} \mod N$.
7. Avec ce secret partagé, on peut envoyer des messages.

SECURITÉ = difficulté de **problème du logarithme discret** :

Données : $g, \overline{A}, N \in \mathbb{N}$, et $\exists a$, $0 < a < N$ t.q. $g^a = \overline{A} \mod N$.

Problème : retrouver a.

Cryprographie via secret partagé, Diffie Hellman.

Compliquée par deux collections d'auteurs (1999): Anshel, Anshel et Goldfeld, et de Ko, Lee, Cheon, Han, Kang et Park.

1. Nombre premier N et $0 < g < N$. Les entiers mod N = groupe cyclique additif.
2. Alice choisit un entier $0 < a < N$, Bob choisit $0 < b < N$.
3. Alice calcule $g^a = \overline{A} \mod N$. Bob calcule $g^b = \overline{B} \mod N$.
4. Alice et Bob échange les resultats $\overline{A}$ et $\overline{B}$.
5. Alice reçoit $\overline{B}$ et avec a, calcule $(\overline{B})^a = g^{(ab)} \mod N$.
6. Bob reçoit $\overline{A}$ et avec b, calcule $(\overline{A})^b = g^{(ba)} \mod N$.
7. Avec ce secret partagé, on peut envoyer des messages.

SECURITÉ = difficulté de **problème du logarithme discret** :

Données : $g, \overline{A}, N \in \mathbb{N}$, et $\exists a$, $0 < a < N$ t.q. $g^a = \overline{A} \mod N$.

Problème : retrouver a.

Cryprographie via secret partagé, Diffie Hellman.

Compliquée par deux collections d'auteurs (1999): Anshel, Anshel et Goldfeld, et de Ko, Lee, Cheon, Han, Kang et Park.

1. Nombre premier N et $0 < g < N$. Les entiers mod N = groupe cyclique additif.
2. Alice choisit un entier $0 < a < N$, Bob choisit $0 < b < N$.
3. Alice calcule $g^a = \overline{A} \mod N$. Bob calcule $g^b = \overline{B} \mod N$.
4. Alice et Bob échange les resultats $\overline{A}$ et $\overline{B}$.
5. Alice reçoit $\overline{B}$ et avec a, calcule $(\overline{B})^a = g^{(ab)} \mod N$.
6. Bob reçoit $\overline{A}$ et avec b, calcule $(\overline{A})^b = g^{(ba)} \mod N$.
7. Avec ce secret partagé, on peut envoyer des messages.

SECURITÉ = difficulté de **problème du logarithme discret** :
Données : $g, \overline{A}, N \in \mathbb{N}$, et $\exists a$, $0 < a < N$ t.q. $g^a = \overline{A} \mod N$.
Problème : retrouver a.

Cryprographie via secret partagé, Diffie Hellman.

Compliquée par deux collections d'auteurs (1999): Anshel, Anshel et Goldfeld, et de Ko, Lee, Cheon, Han, Kang et Park.

1. Nombre premier N et $0 < g < N$. Les entiers mod N = groupe cyclique additif.
2. Alice choisit un entier $0 < a < N$, Bob choisit $0 < b < N$.
3. Alice calcule $g^a = \overline{A} \mod N$. Bob calcule $g^b = \overline{B} \mod N$.
4. Alice et Bob échange les resultats $\overline{A}$ et $\overline{B}$.
5. Alice reçoit $\overline{B}$ et avec a, calcule $(\overline{B})^a = g^{(ab)} \mod N$.
6. Bob reçoit $\overline{A}$ et avec b, calcule $(\overline{A})^b = g^{(ba)} \mod N$.
7. Avec ce secret partagé, on peut envoyer des messages.

SECURITÉ = difficulté de **problème du logarithme discret** :

Données : $g, \overline{A}, N \in \mathbb{N}$, et $\exists a$, $0 < a < N$ t.q. $g^a = \overline{A} \mod N$.

Problème : retrouver a.

Cryprographie via secret partagé, Diffie Hellman.

Compliquée par deux collections d'auteurs (1999): Anshel, Anshel et Goldfeld, et de Ko, Lee, Cheon, Han, Kang et Park.

1. Nombre premier N et $0 < g < N$. Les entiers mod N = groupe cyclique additif.
2. Alice choisit un entier $0 < a < N$, Bob choisit $0 < b < N$.
3. Alice calcule $g^a = \overline{A} \mod N$. Bob calcule $g^b = \overline{B} \mod N$.
4. Alice et Bob échange les resultats $\overline{A}$ et $\overline{B}$.
5. Alice reçoit $\overline{B}$ et avec a, calcule $(\overline{B})^a = g^{(ab)} \mod N$.
6. Bob reçoit $\overline{A}$ et avec b, calcule $(\overline{A})^b = g^{(ba)} \mod N$.
7. Avec ce secret partagé, on peut envoyer des messages.

SECURITÉ = difficulté de **problème du logarithme discret** :

Données : $g, \overline{A}, N \in \mathbb{N}$, et $\exists a$, $0 < a < N$ t.q. $g^a = \overline{A} \mod N$.

Problème : retrouver a.

Cryprographie via secret partagé, Diffie Hellman.

Compliquée par deux collections d'auteurs (1999): Anshel, Anshel et Goldfeld, et de Ko, Lee, Cheon, Han, Kang et Park.

1. Nombre premier N et $0 < g < N$. Les entiers mod N = groupe cyclique additif.
2. Alice choisit un entier $0 < a < N$, Bob choisit $0 < b < N$.
3. Alice calcule $g^a = \overline{A} \mod N$. Bob calcule $g^b = \overline{B} \mod N$.
4. Alice et Bob échange les resultats $\overline{A}$ et $\overline{B}$.
5. Alice reçoit $\overline{B}$ et avec a, calcule $(\overline{B})^a = g^{(ab)} \mod N$.
6. Bob reçoit $\overline{A}$ et avec b, calcule $(\overline{A})^b = g^{(ba)} \mod N$.
7. Avec ce secret partagé, on peut envoyer des messages. comment coder

SECURITÉ = difficulté de **problème du logarithme discret** :

Données : $g, \overline{A}, N \in \mathbb{N}$, et $\exists a$, $0 < a < N$ t.q. $g^a = \overline{A} \mod N$.

Problème : retrouver a.

Cryprographie via secret partagé, Diffie Hellman.

Compliquée par deux collections d'auteurs (1999): Anshel, Anshel et Goldfeld, et de Ko, Lee, Cheon, Han, Kang et Park.

1. Nombre premier N et $0 < g < N$. Les entiers mod N = groupe cyclique additif.
2. Alice choisit un entier $0 < a < N$, Bob choisit $0 < b < N$.
3. Alice calcule $g^a = \overline{A} \mod N$. Bob calcule $g^b = \overline{B} \mod N$.
4. Alice et Bob échange les resultats $\overline{A}$ et $\overline{B}$.
5. Alice reçoit $\overline{B}$ et avec a, calcule $(\overline{B})^a = g^{(ab)} \mod N$.
6. Bob reçoit $\overline{A}$ et avec b, calcule $(\overline{A})^b = g^{(ba)} \mod N$.
7. Avec ce secret partagé, on peut envoyer des messages. comment coder

SECURITÉ = difficulté de **problème du logarithme discret** :

Données : $g, \overline{A}, N \in \mathbb{N}$, et $\exists a$, $0 < a < N$ t.q. $g^a = \overline{A} \mod N$.

Problème : retrouver a.

Cryprographie via secret partagé, Diffie Hellman.

Compliquée par deux collections d'auteurs (1999): Anshel, Anshel et Goldfeld, et de Ko, Lee, Cheon, Han, Kang et Park.

1. Nombre premier N et $0 < g < N$. Les entiers mod N = groupe cyclique additif.
2. Alice choisit un entier $0 < a < N$, Bob choisit $0 < b < N$.
3. Alice calcule $g^a = \overline{A} \mod N$. Bob calcule $g^b = \overline{B} \mod N$.
4. Alice et Bob échange les resultats $\overline{A}$ et $\overline{B}$.
5. Alice reçoit $\overline{B}$ et avec a, calcule $(\overline{B})^a = g^{(ab)} \mod N$.
6. Bob reçoit $\overline{A}$ et avec b, calcule $(\overline{A})^b = g^{(ba)} \mod N$.
7. Avec ce secret partagé, on peut envoyer des messages. comment coder

SECURITÉ = difficulté de **problème du logarithme discret** :

Données : $g, \overline{A}, N \in \mathbb{N}$, et $\exists a$, $0 < a < N$ t.q. $g^a = \overline{A} \mod N$.

Problème : retrouver a.

Cryprographie via secret partagé, Diffie Hellman.

Compliquée par deux collections d'auteurs (1999): Anshel, Anshel et Goldfeld, et de Ko, Lee, Cheon, Han, Kang et Park.

1. Nombre premier N et $0 < g < N$. Les entiers mod N = groupe cyclique additif.
2. Alice choisit un entier $0 < a < N$, Bob choisit $0 < b < N$.
3. Alice calcule $g^a = \overline{A} \mod N$. Bob calcule $g^b = \overline{B} \mod N$.
4. Alice et Bob échange les resultats $\overline{A}$ et $\overline{B}$.
5. Alice reçoit $\overline{B}$ et avec a, calcule $(\overline{B})^a = g^{(ab)} \mod N$.
6. Bob reçoit $\overline{A}$ et avec b, calcule $(\overline{A})^b = g^{(ba)} \mod N$.
7. Avec ce secret partagé, on peut envoyer des messages. ▸ comment coder

SECURITÉ = difficulté de **problème du logarithme discret** :

Données : $g, \overline{A}, N \in \mathbb{N}$, et $\exists a$, $0 < a < N$ t.q. $g^a = \overline{A} \mod N$.

Problème : retrouver a.

Cryprographie via secret partagé, Diffie Hellman.

Compliquée par deux collections d'auteurs (1999): Anshel, Anshel et Goldfeld, et de Ko, Lee, Cheon, Han, Kang et Park.

1. Nombre premier N et $0 < g < N$. Les entiers mod N = groupe cyclique additif.
2. Alice choisit un entier $0 < a < N$, Bob choisit $0 < b < N$.
3. Alice calcule $g^a = \overline{A} \mod N$. Bob calcule $g^b = \overline{B} \mod N$.
4. Alice et Bob échange les resultats $\overline{A}$ et $\overline{B}$.
5. Alice reçoit $\overline{B}$ et avec a, calcule $(\overline{B})^a = g^{(ab)} \mod N$.
6. Bob reçoit $\overline{A}$ et avec b, calcule $(\overline{A})^b = g^{(ba)} \mod N$.
7. Avec ce secret partagé, on peut envoyer des messages. ▸ comment coder

SECURITÉ = difficulté de **problème du logarithme discret** :

Données : $g, \overline{A}, N \in \mathbb{N}$, et $\exists a$, $0 < a < N$ t.q. $g^a = \overline{A} \mod N$.

Problème : retrouver a.

Cryprographie via secret partagé, Diffie Hellman.

Compliquée par deux collections d'auteurs (1999): Anshel, Anshel et Goldfeld, et de Ko, Lee, Cheon, Han, Kang et Park.

1. Nombre premier N et $0 < g < N$. Les entiers mod N = groupe cyclique additif.
2. Alice choisit un entier $0 < a < N$, Bob choisit $0 < b < N$.
3. Alice calcule $g^a = \overline{A} \mod N$. Bob calcule $g^b = \overline{B} \mod N$.
4. Alice et Bob échange les resultats $\overline{A}$ et $\overline{B}$.
5. Alice reçoit $\overline{B}$ et avec a, calcule $(\overline{B})^a = g^{(ab)} \mod N$.
6. Bob reçoit $\overline{A}$ et avec b, calcule $(\overline{A})^b = g^{(ba)} \mod N$.
7. Avec ce secret partagé, on peut envoyer des messages. ▸ comment coder

SECURITÉ = difficulté de **problème du logarithme discret** :

Données : $g, \overline{A}, N \in \mathbb{N}$, et $\exists a$, $0 < a < N$ t.q. $g^a = \overline{A} \mod N$.

Problème : retrouver a.

Cryprographie via secret partagé, Diffie Hellman.

Compliquée par deux collections d'auteurs (1999): Anshel, Anshel et Goldfeld, et de Ko, Lee, Cheon, Han, Kang et Park.

1. Nombre premier N et $0 < g < N$. Les entiers mod N = groupe cyclique additif.
2. Alice choisit un entier $0 < a < N$, Bob choisit $0 < b < N$.
3. Alice calcule $g^a = \overline{A} \mod N$. Bob calcule $g^b = \overline{B} \mod N$.
4. Alice et Bob échange les resultats $\overline{A}$ et $\overline{B}$.
5. Alice reçoit $\overline{B}$ et avec a, calcule $(\overline{B})^a = g^{(ab)} \mod N$.
6. Bob reçoit $\overline{A}$ et avec b, calcule $(\overline{A})^b = g^{(ba)} \mod N$.
7. Avec ce secret partagé, on peut envoyer des messages. ▸ comment coder

SECURITÉ = difficulté de **problème du logarithme discret** :

Données : $g, \overline{A}, N \in \mathbb{N}$, et $\exists a$, $0 < a < N$ t.q. $g^a = \overline{A} \mod N$.

Problème : retrouver a.

Version 2.0 (AAG): $SL(2,\mathbb{Z})$, matrices inversibles 2×2, entrées entiers

1. Alice publie une liste $X_A = \{M_1, \ldots, M_m\}$ de quelques matrices.
   Bob publie une liste $X_B = \{N_1, \ldots, N_n\}$ de quelques matrices.
2. Bob choisit un produit $Q(X_A)$ dans les matrices de Alice :
   ex $Q(X_A) = M_2{}^3 M_1{}^{-2} M_3 M_1{}^{-1} M_3$
   Alice choisit un produit $P(X_B)$ dans les matrices de Bob.
3. Alice calcule, et envoie $PX_AP^{-1} = \{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$.
   Bob envoie $QX_BQ^{-1} = \{QN_1Q^{-1}, \ldots, QN_mQ^{-1}\}$.
4. Bob recalcule son $Q(X_A)$ en termes de $\{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$:
   ex: $(PM_2P^{-1})^3(PM_1P^{-1})^{-2}(PM_3P^{-1})(PM_1P^{-1})^{-1}PM_3P^{-1}$
   $= P(M_2{}^3 M_1{}^{-2} M_3 M_1{}^{-1} M_3)P^{-1} = PQ(X_A)P^{-1}$
   Bob peut aussi calculer Q^{-1} et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
5. Alice recalcule son $P(X_B)$ en termes de $\{QN_1Q^{-1}, \ldots, QN_nQ^{-1}\}$. Et comme avant $P(QX_BQ^{-1}) = QP(X_B)Q^{-1}$.
   Alice peut prendre l'inverse $QP^{-1}Q^{-1}$ et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
6. Clé secrète partagée : $PQP^{-1}Q^{-1}$.

Version 2.0 (AAG): $SL(2,\mathbb{Z})$, matrices inversibles 2×2, entrées entiers

1. Alice publie une liste $X_A = \{M_1, \ldots, M_m\}$ de quelques matrices.
   Bob publie une liste $X_B = \{N_1, \ldots, N_n\}$ de quelques matrices.
2. Bob choisit un produit $Q(X_A)$ dans les matrices de Alice :
   ex $Q(X_A) = {M_2}^3 {M_1}^{-2} M_3 {M_1}^{-1} M_3$
   Alice choisit un produit $P(X_B)$ dans les matrices de Bob.
3. Alice calcule, et envoie $PX_AP^{-1} = \{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$.
   Bob envoie $QX_BQ^{-1} = \{QN_1Q^{-1}, \ldots, QN_mQ^{-1}\}$.
4. Bob recalcule son $Q(X_A)$ en termes de $\{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$:
   ex: $(PM_2P^{-1})^3(PM_1P^{-1})^{-2}(PM_3P^{-1})(PM_1P^{-1})^{-1}PM_3P^{-1}$
   $= P({M_2}^3 {M_1}^{-2} M_3 {M_1}^{-1} M_3)P^{-1} = PQ(X_A)P^{-1}$
   Bob peut aussi calculer Q^{-1} et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
5. Alice recalcule son $P(X_B)$ en termes de $\{QN_1Q^{-1}, \ldots, QN_nQ^{-1}\}$. Et comme avant $P(QX_BQ^{-1}) = QP(X_B)Q^{-1}$.
   Alice peut prendre l'inverse $QP^{-1}Q^{-1}$ et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
6. Clé secrète partagée : $PQP^{-1}Q^{-1}$.

Version 2.0 (AAG): $SL(2,\mathbb{Z})$, matrices inversibles 2×2, entrées entiers

1. Alice publie une liste $X_A = \{M_1, \ldots, M_m\}$ de quelques matrices.
   Bob publie une liste $X_B = \{N_1, \ldots, N_n\}$ de quelques matrices.
2. Bob choisit un produit $Q(X_A)$ dans les matrices de Alice :
   ex $Q(X_A) = M_2{}^3 M_1{}^{-2} M_3 M_1{}^{-1} M_3$
   Alice choisit un produit $P(X_B)$ dans les matrices de Bob.
3. Alice calcule, et envoie $PX_AP^{-1} = \{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$.
   Bob envoie $QX_BQ^{-1} = \{QN_1Q^{-1}, \ldots, QN_mQ^{-1}\}$.
4. Bob recalcule son $Q(X_A)$ en termes de $\{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$:
   ex: $(PM_2P^{-1})^3(PM_1P^{-1})^{-2}(PM_3P^{-1})(PM_1P^{-1})^{-1}PM_3P^{-1}$
   $= P(M_2{}^3 M_1{}^{-2} M_3 M_1{}^{-1} M_3)P^{-1} = PQ(X_A)P^{-1}$
   Bob peut aussi calculer Q^{-1} et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
5. Alice recalcule son $P(X_B)$ en termes de $\{QN_1Q^{-1}, \ldots, QN_nQ^{-1}\}$. Et comme avant $P(QX_BQ^{-1}) = QP(X_B)Q^{-1}$.
   Alice peut prendre l'inverse $QP^{-1}Q^{-1}$ et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
6. Clé secrète partagée : $PQP^{-1}Q^{-1}$.

Version 2.0 (AAG): $SL(2,\mathbb{Z})$, matrices inversibles 2×2, entrées entiers

1. Alice publie une liste $X_A = \{M_1, \ldots, M_m\}$ de quelques matrices.
   Bob publie une liste $X_B = \{N_1, \ldots, N_n\}$ de quelques matrices.
2. Bob choisit un produit $Q(X_A)$ dans les matrices de Alice :
   ex $Q(X_A) = {M_2}^3 {M_1}^{-2} M_3 {M_1}^{-1} M_3$
   Alice choisit un produit $P(X_B)$ dans les matrices de Bob.
3. Alice calcule, et envoie $PX_AP^{-1} = \{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$.
   Bob envoie $QX_BQ^{-1} = \{QN_1Q^{-1}, \ldots, QN_mQ^{-1}\}$.
4. Bob recalcule son $Q(X_A)$ en termes de $\{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$:
   ex: $(PM_2P^{-1})^3(PM_1P^{-1})^{-2}(PM_3P^{-1})(PM_1P^{-1})^{-1}PM_3P^{-1}$
   $= P({M_2}^3 {M_1}^{-2} M_3 {M_1}^{-1} M_3)P^{-1} = PQ(X_A)P^{-1}$
   Bob peut aussi calculer Q^{-1} et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
5. Alice recalcule son $P(X_B)$ en termes de $\{QN_1Q^{-1}, \ldots, QN_nQ^{-1}\}$. Et comme avant $P(QX_BQ^{-1}) = QP(X_B)Q^{-1}$.
   Alice peut prendre l'inverse $QP^{-1}Q^{-1}$ et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
6. Clé secrète partagée : $PQP^{-1}Q^{-1}$.

Version 2.0 (AAG): $SL(2,\mathbb{Z})$, matrices inversibles 2×2, entrées entiers

1. Alice publie une liste $X_A = \{M_1, \ldots, M_m\}$ de quelques matrices.
   Bob publie une liste $X_B = \{N_1, \ldots, N_n\}$ de quelques matrices.
2. Bob choisit un produit $Q(X_A)$ dans les matrices de Alice :
   ex $Q(X_A) = {M_2}^3 {M_1}^{-2} M_3 {M_1}^{-1} M_3$
   Alice choisit un produit $P(X_B)$ dans les matrices de Bob.
3. Alice calcule, et envoie $PX_AP^{-1} = \{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$.
   Bob envoie $QX_BQ^{-1} = \{QN_1Q^{-1}, \ldots, QN_mQ^{-1}\}$.
4. Bob recalcule son $Q(X_A)$ en termes de $\{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$:
   ex: $(PM_2P^{-1})^3(PM_1P^{-1})^{-2}(PM_3P^{-1})(PM_1P^{-1})^{-1}PM_3P^{-1}$
   $= P({M_2}^3 {M_1}^{-2} M_3 {M_1}^{-1} M_3)P^{-1} = PQ(X_A)P^{-1}$
   Bob peut aussi calculer Q^{-1} et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
5. Alice recalcule son $P(X_B)$ en termes de $\{QN_1Q^{-1}, \ldots, QN_nQ^{-1}\}$. Et comme avant $P(QX_BQ^{-1}) = QP(X_B)Q^{-1}$.
   Alice peut prendre l'inverse $QP^{-1}Q^{-1}$ et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
6. Clé secrète partagée : $PQP^{-1}Q^{-1}$.

Version 2.0 (AAG): $SL(2,\mathbb{Z})$, matrices inversibles 2×2, entrées entiers

1. Alice publie une liste $X_A = \{M_1, \ldots, M_m\}$ de quelques matrices.
   Bob publie une liste $X_B = \{N_1, \ldots, N_n\}$ de quelques matrices.
2. Bob choisit un produit $Q(X_A)$ dans les matrices de Alice :
   ex $Q(X_A) = {M_2}^3 {M_1}^{-2} M_3 {M_1}^{-1} M_3$
   Alice choisit un produit $P(X_B)$ dans les matrices de Bob.
3. Alice calcule, et envoie $PX_AP^{-1} = \{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$.
   Bob envoie $QX_BQ^{-1} = \{QN_1Q^{-1}, \ldots, QN_mQ^{-1}\}$.
4. Bob recalcule son $Q(X_A)$ en termes de $\{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$:
   ex: $(PM_2P^{-1})^3(PM_1P^{-1})^{-2}(PM_3P^{-1})(PM_1P^{-1})^{-1}PM_3P^{-1}$
   $= P({M_2}^3 {M_1}^{-2} M_3 {M_1}^{-1} M_3)P^{-1} = PQ(X_A)P^{-1}$
   Bob peut aussi calculer Q^{-1} et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
5. Alice recalcule son $P(X_B)$ en termes de $\{QN_1Q^{-1}, \ldots, QN_nQ^{-1}\}$. Et comme avant $P(QX_BQ^{-1}) = QP(X_B)Q^{-1}$.
   Alice peut prendre l'inverse $QP^{-1}Q^{-1}$ et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
6. Clé secrète partagée : $PQP^{-1}Q^{-1}$.

Version 2.0 (AAG): $SL(2,\mathbb{Z})$, matrices inversibles 2×2, entrées entiers

1. Alice publie une liste $X_A = \{M_1, \ldots, M_m\}$ de quelques matrices.
   Bob publie une liste $X_B = \{N_1, \ldots, N_n\}$ de quelques matrices.
2. Bob choisit un produit $Q(X_A)$ dans les matrices de Alice :
   ex $Q(X_A) = {M_2}^3 {M_1}^{-2} M_3 {M_1}^{-1} M_3$
   Alice choisit un produit $P(X_B)$ dans les matrices de Bob.
3. Alice calcule, et envoie $PX_AP^{-1} = \{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$.
   Bob envoie $QX_BQ^{-1} = \{QN_1Q^{-1}, \ldots, QN_mQ^{-1}\}$.
4. Bob recalcule son $Q(X_A)$ en termes de $\{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$:
   ex: $(PM_2P^{-1})^3(PM_1P^{-1})^{-2}(PM_3P^{-1})(PM_1P^{-1})^{-1}PM_3P^{-1}$
   $= P({M_2}^3 {M_1}^{-2} M_3 {M_1}^{-1} M_3)P^{-1} = PQ(X_A)P^{-1}$
   Bob peut aussi calculer Q^{-1} et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
5. Alice recalcule son $P(X_B)$ en termes de $\{QN_1Q^{-1}, \ldots, QN_nQ^{-1}\}$. Et comme avant $P(QX_BQ^{-1}) = QP(X_B)Q^{-1}$.
   Alice peut prendre l'inverse $QP^{-1}Q^{-1}$ et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
6. Clé secrète partagée : $PQP^{-1}Q^{-1}$.

Version 2.0 (AAG): $SL(2,\mathbb{Z})$, matrices inversibles 2×2, entrées entiers

1. Alice publie une liste $X_A = \{M_1, \ldots, M_m\}$ de quelques matrices.
   Bob publie une liste $X_B = \{N_1, \ldots, N_n\}$ de quelques matrices.
2. Bob choisit un produit $Q(X_A)$ dans les matrices de Alice :
   ex $Q(X_A) = {M_2}^3 {M_1}^{-2} M_3 {M_1}^{-1} M_3$
   Alice choisit un produit $P(X_B)$ dans les matrices de Bob.
3. Alice calcule, et envoie $PX_AP^{-1} = \{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$.
   Bob envoie $QX_BQ^{-1} = \{QN_1Q^{-1}, \ldots, QN_mQ^{-1}\}$.
4. Bob recalcule son $Q(X_A)$ en termes de $\{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$:
   ex: $(PM_2P^{-1})^3(PM_1P^{-1})^{-2}(PM_3P^{-1})(PM_1P^{-1})^{-1}PM_3P^{-1}$
   $= P({M_2}^3 {M_1}^{-2} M_3 {M_1}^{-1} M_3)P^{-1} = PQ(X_A)P^{-1}$
   Bob peut aussi calculer Q^{-1} et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
5. Alice recalcule son $P(X_B)$ en termes de $\{QN_1Q^{-1}, \ldots, QN_nQ^{-1}\}$. Et comme avant $P(QX_BQ^{-1}) = QP(X_B)Q^{-1}$.
   Alice peut prendre l'inverse $QP^{-1}Q^{-1}$ et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
6. Clé secrète partagée : $PQP^{-1}Q^{-1}$.

Version 2.0 (AAG): $SL(2,\mathbb{Z})$, matrices inversibles 2×2, entrées entiers

1. Alice publie une liste $X_A = \{M_1, \ldots, M_m\}$ de quelques matrices.
   Bob publie une liste $X_B = \{N_1, \ldots, N_n\}$ de quelques matrices.
2. Bob choisit un produit $Q(X_A)$ dans les matrices de Alice :
   ex $Q(X_A) = {M_2}^3 {M_1}^{-2} M_3 {M_1}^{-1} M_3$
   Alice choisit un produit $P(X_B)$ dans les matrices de Bob.
3. Alice calcule, et envoie $PX_AP^{-1} = \{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$.
   Bob envoie $QX_BQ^{-1} = \{QN_1Q^{-1}, \ldots, QN_mQ^{-1}\}$.
4. Bob recalcule son $Q(X_A)$ en termes de $\{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$:
   ex: $(PM_2P^{-1})^3(PM_1P^{-1})^{-2}(PM_3P^{-1})(PM_1P^{-1})^{-1}PM_3P^{-1}$
   $= P({M_2}^3 {M_1}^{-2} M_3 {M_1}^{-1} M_3)P^{-1} = PQ(X_A)P^{-1}$
   Bob peut aussi calculer Q^{-1} et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
5. Alice recalcule son $P(X_B)$ en termes de $\{QN_1Q^{-1}, \ldots, QN_nQ^{-1}\}$. Et comme avant $P(QX_BQ^{-1}) = QP(X_B)Q^{-1}$.
   Alice peut prendre l'inverse $QP^{-1}Q^{-1}$ et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
6. Clé secrète partagée : $PQP^{-1}Q^{-1}$.

Version 2.0 (AAG): $SL(2,\mathbb{Z})$, matrices inversibles 2×2, entrées entiers

1. Alice publie une liste $X_A = \{M_1, \ldots, M_m\}$ de quelques matrices.
   Bob publie une liste $X_B = \{N_1, \ldots, N_n\}$ de quelques matrices.
2. Bob choisit un produit $Q(X_A)$ dans les matrices de Alice :
   ex $Q(X_A) = {M_2}^3 {M_1}^{-2} M_3 {M_1}^{-1} M_3$
   Alice choisit un produit $P(X_B)$ dans les matrices de Bob.
3. Alice calcule, et envoie $PX_AP^{-1} = \{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$.
   Bob envoie $QX_BQ^{-1} = \{QN_1Q^{-1}, \ldots, QN_mQ^{-1}\}$.
4. Bob recalcule son $Q(X_A)$ en termes de $\{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$:
   ex: $(PM_2P^{-1})^3(PM_1P^{-1})^{-2}(PM_3P^{-1})(PM_1P^{-1})^{-1}PM_3P^{-1}$
   $= P({M_2}^3 {M_1}^{-2} M_3 {M_1}^{-1} M_3)P^{-1} = PQ(X_A)P^{-1}$
   Bob peut aussi calculer Q^{-1} et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
5. Alice recalcule son $P(X_B)$ en termes de $\{QN_1Q^{-1}, \ldots, QN_nQ^{-1}\}$. Et comme avant $P(QX_BQ^{-1}) = QP(X_B)Q^{-1}$.
   Alice peut prendre l'inverse $QP^{-1}Q^{-1}$ et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
6. Clé secrète partagée : $PQP^{-1}Q^{-1}$.

Version 2.0 (AAG): $SL(2,\mathbb{Z})$, matrices inversibles 2×2, entrées entiers

1. Alice publie une liste $X_A = \{M_1, \ldots, M_m\}$ de quelques matrices.
   Bob publie une liste $X_B = \{N_1, \ldots, N_n\}$ de quelques matrices.
2. Bob choisit un produit $Q(X_A)$ dans les matrices de Alice :
   ex $Q(X_A) = {M_2}^3 {M_1}^{-2} M_3 {M_1}^{-1} M_3$
   Alice choisit un produit $P(X_B)$ dans les matrices de Bob.
3. Alice calcule, et envoie $PX_AP^{-1} = \{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$.
   Bob envoie $QX_BQ^{-1} = \{QN_1Q^{-1}, \ldots, QN_mQ^{-1}\}$.
4. Bob recalcule son $Q(X_A)$ en termes de $\{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$:
   ex: $(PM_2P^{-1})^3(PM_1P^{-1})^{-2}(PM_3P^{-1})(PM_1P^{-1})^{-1}PM_3P^{-1}$
   $= P({M_2}^3 {M_1}^{-2} M_3 {M_1}^{-1} M_3)P^{-1} = PQ(X_A)P^{-1}$
   Bob peut aussi calculer Q^{-1} et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
5. Alice recalcule son $P(X_B)$ en termes de $\{QN_1Q^{-1}, \ldots, QN_nQ^{-1}\}$. Et comme avant $P(QX_BQ^{-1}) = QP(X_B)Q^{-1}$.
   Alice peut prendre l'inverse $QP^{-1}Q^{-1}$ et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
6. Clé secrète partagée : $PQP^{-1}Q^{-1}$.

Version 2.0 (AAG): $SL(2,\mathbb{Z})$, matrices inversibles 2×2, entrées entiers

1. Alice publie une liste $X_A = \{M_1, \ldots, M_m\}$ de quelques matrices.
   Bob publie une liste $X_B = \{N_1, \ldots, N_n\}$ de quelques matrices.
2. Bob choisit un produit $Q(X_A)$ dans les matrices de Alice :
   ex $Q(X_A) = {M_2}^3 {M_1}^{-2} M_3 {M_1}^{-1} M_3$
   Alice choisit un produit $P(X_B)$ dans les matrices de Bob.
3. Alice calcule, et envoie $PX_AP^{-1} = \{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$.
   Bob envoie $QX_BQ^{-1} = \{QN_1Q^{-1}, \ldots, QN_mQ^{-1}\}$.
4. Bob recalcule son $Q(X_A)$ en termes de $\{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$:
   ex: $(PM_2P^{-1})^3(PM_1P^{-1})^{-2}(PM_3P^{-1})(PM_1P^{-1})^{-1}PM_3P^{-1}$
   $= P({M_2}^3 {M_1}^{-2} M_3 {M_1}^{-1} M_3)P^{-1} = PQ(X_A)P^{-1}$
   Bob peut aussi calculer Q^{-1} et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
5. Alice recalcule son $P(X_B)$ en termes de $\{QN_1Q^{-1}, \ldots, QN_nQ^{-1}\}$. Et comme avant $P(QX_BQ^{-1}) = QP(X_B)Q^{-1}$.
   Alice peut prendre l'inverse $QP^{-1}Q^{-1}$ et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
6. Clé secrète partagée : $PQP^{-1}Q^{-1}$.

Version 2.0 (AAG): $SL(2,\mathbb{Z})$, matrices inversibles 2×2, entrées entiers

1. Alice publie une liste $X_A = \{M_1, \ldots, M_m\}$ de quelques matrices.
   Bob publie une liste $X_B = \{N_1, \ldots, N_n\}$ de quelques matrices.
2. Bob choisit un produit $Q(X_A)$ dans les matrices de Alice :
   ex $Q(X_A) = {M_2}^3 {M_1}^{-2} M_3 {M_1}^{-1} M_3$
   Alice choisit un produit $P(X_B)$ dans les matrices de Bob.
3. Alice calcule, et envoie $PX_AP^{-1} = \{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$.
   Bob envoie $QX_BQ^{-1} = \{QN_1Q^{-1}, \ldots, QN_mQ^{-1}\}$.
4. Bob recalcule son $Q(X_A)$ en termes de $\{PM_1P^{-1}, \ldots, PM_mP^{-1}\}$:
   ex: $(PM_2P^{-1})^3(PM_1P^{-1})^{-2}(PM_3P^{-1})(PM_1P^{-1})^{-1}PM_3P^{-1}$
   $= P({M_2}^3 {M_1}^{-2} M_3 {M_1}^{-1} M_3)P^{-1} = PQ(X_A)P^{-1}$
   Bob peut aussi calculer Q^{-1} et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
5. Alice recalcule son $P(X_B)$ en termes de $\{QN_1Q^{-1}, \ldots, QN_nQ^{-1}\}$. Et comme avant $P(QX_BQ^{-1}) = QP(X_B)Q^{-1}$.
   Alice peut prendre l'inverse $QP^{-1}Q^{-1}$ et donc calculer $\mathbf{PQP^{-1}Q^{-1}}$.
6. Clé secrète partagée : $PQP^{-1}Q^{-1}$.

MAIS: on peut retrouver P et Q de l'information publique!
MÊME si on utilise $GL(10, \mathbb{Q})$ ou autre groupe de matrices.
Donc il faut un domaine de calcul plus compliqué.

Les groupes infinis non–abéliens ("crash course")

La théorie des groupes infinis non–abéliens donne des exemples des groupes dans lesquels certains problèmes, en particulier le problème de conjugaison, n'ont pas de solution algorithmique, ou ont des solutions dificiles, avec des démonstrations de la difficulté.

Definition

Un groupe est un ensemble avec une opération associative t.q. il existe un élément neutre et chaque élément possède un inverse.

Exemples: $(\mathbb{Z}, +)$, $(\mathbb{R}^{>0}, \times)$ $(\frac{\mathbb{Z}}{N\mathbb{Z}}, +)$, $(\mathbb{Z} \times \mathbb{Z}, +)$, S_n et $(GL(n, \mathbb{Q}), \times)$.

MAIS: on peut retrouver P et Q de l'information publique!
MÊME si on utilise $GL(10, \mathbb{Q})$ ou autre groupe de matrices.
Donc il faut un domaine de calcul plus compliqué.

Les groupes infinis non–abéliens ("crash course")

La théorie des groupes infinis non–abéliens donne des exemples des groupes dans lesquels certains problèmes, en particulier le problème de conjugaison, n'ont pas de solution algorithmique, ou ont des solutions dificiles, avec des démonstrations de la difficulté.

Definition

Un groupe est un ensemble avec une opération associative t.q. il existe un élément neutre et chaque élément possède un inverse.

Exemples: $(\mathbb{Z}, +)$, $(\mathbb{R}^{>0}, \times)$ $(\frac{\mathbb{Z}}{N\mathbb{Z}}, +)$, $(\mathbb{Z} \times \mathbb{Z}, +)$, S_n et $(GL(n, \mathbb{Q}), \times)$.

MAIS: on peut retrouver P et Q de l'information publique!
MÊME si on utilise $GL(10, \mathbb{Q})$ ou autre groupe de matrices.
Donc il faut un domaine de calcul plus compliqué.

Les groupes infinis non–abéliens ("crash course")

La théorie des groupes infinis non–abéliens donne des exemples des groupes dans lesquels certains problèmes, en particulier le problème de conjugaison, n'ont pas de solution algorithmique, ou ont des solutions dificiles, avec des démonstrations de la difficulté.

Definition

Un groupe est un ensemble avec une opération associative t.q. il existe un élément neutre et chaque élément possède un inverse.

Exemples: $(\mathbb{Z}, +)$, $(\mathbb{R}^{>0}, \times)$ $(\frac{\mathbb{Z}}{N\mathbb{Z}}, +)$, $(\mathbb{Z} \times \mathbb{Z}, +)$, S_n et $(GL(n, \mathbb{Q}), \times)$.

MAIS: on peut retrouver P et Q de l'information publique!
MÊME si on utilise $GL(10, \mathbb{Q})$ ou autre groupe de matrices.
Donc il faut un domaine de calcul plus compliqué.

Les groupes infinis non–abéliens ("crash course")

La théorie des groupes infinis non–abéliens donne des exemples des groupes dans lesquels certains problèmes, en particulier le problème de conjugaison, n'ont pas de solution algorithmique, ou ont des solutions dificiles, avec des démonstrations de la difficulté.

Definition

Un groupe est un ensemble avec une opération associative t.q. il existe un élément neutre et chaque élément possède un inverse.

Exemples: $(\mathbb{Z}, +)$, $(\mathbb{R}^{>0}, \times)$ $(\frac{\mathbb{Z}}{N\mathbb{Z}}, +)$, $(\mathbb{Z} \times \mathbb{Z}, +)$, S_n et $(GL(n, \mathbb{Q}), \times)$.

Le groupe libre $F(a, b)$ sur $\{a, b\}$

L'ensemble est tous les "mots" en a, b, a^{-1}, b^{-1} modulo une relation;

L'opération est concatenation des mots;
l'élément neutre est 1 qui est le mot vide;
l'inverse de a est a^{-1}, inverse de b est b^{-1}

Modulo la relation que $1 = aa^{-1} = a^{-1}a = bb^{-1} = b^{-1}b$.

Donc $(abbba^{-1}a^{-1}b^{-1}b^{-1}b^{-1})(bbbaab^{-1}) = abb$
et $(abbba^{-1}a^{-1}b^{-1}b^{-1}b^{-1})^{-1} = bbbaab^{-1}b^{-1}b^{-1}a^{-1}$

En matrices on peut prendre $a = \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}$ et $b = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}$

On peut faire la même chose avec autant de générateurs qu'on veut.

Le groupe libre $F(a, b)$ sur $\{a, b\}$

L'ensemble est tous les "mots" en a, b, a^{-1}, b^{-1} modulo une relation;

L'opération est concatenation des mots;
l'élément neutre est 1 qui est le mot vide;
l'inverse de a est a^{-1}, inverse de b est b^{-1}

Modulo la relation que $1 = aa^{-1} = a^{-1}a = bb^{-1} = b^{-1}b$.

Donc $(abbba^{-1}a^{-1}b^{-1}b^{-1}b^{-1})(bbbaab^{-1}) = abb$
et $(abbba^{-1}a^{-1}b^{-1}b^{-1}b^{-1})^{-1} = bbbaab^{-1}b^{-1}b^{-1}a^{-1}$

En matrices on peut prendre $a = \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}$ et $b = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}$

On peut faire la même chose avec autant de générateurs qu'on veut.

Le groupe libre $F(a, b)$ sur $\{a, b\}$

L'ensemble est tous les "mots" en a, b, a^{-1}, b^{-1} modulo une relation;

L'opération est concatenation des mots;
l'élément neutre est 1 qui est le mot vide;
l'inverse de a est a^{-1}, inverse de b est b^{-1}

Modulo la relation que $1 = aa^{-1} = a^{-1}a = bb^{-1} = b^{-1}b$.

Donc $(abbba^{-1}a^{-1}b^{-1}b^{-1}b^{-1})(bbbaab^{-1}) = abb$
et $(abbba^{-1}a^{-1}b^{-1}b^{-1}b^{-1})^{-1} = bbbaab^{-1}b^{-1}b^{-1}a^{-1}$

En matrices on peut prendre $a = \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}$ et $b = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}$

On peut faire la même chose avec autant de générateurs qu'on veut.

Le groupe libre $F(a, b)$ sur $\{a, b\}$

L'ensemble est tous les "mots" en a, b, a^{-1}, b^{-1} modulo une relation;

L'opération est concatenation des mots;
l'élément neutre est 1 qui est le mot vide;
l'inverse de a est a^{-1}, inverse de b est b^{-1}

Modulo la relation que $1 = aa^{-1} = a^{-1}a = bb^{-1} = b^{-1}b$.

Donc $(abbba^{-1}a^{-1}b^{-1}b^{-1}b^{-1})(bbbaab^{-1}) = abb$
et $(abbba^{-1}a^{-1}b^{-1}b^{-1}b^{-1})^{-1} = bbbaab^{-1}b^{-1}b^{-1}a^{-1}$

En matrices on peut prendre $a = \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}$ et $b = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}$

On peut faire la même chose avec autant de générateurs qu'on veut.

Le groupe libre $F(a, b)$ sur $\{a, b\}$

L'ensemble est tous les "mots" en a, b, a^{-1}, b^{-1} modulo une relation;

L'opération est concatenation des mots;
l'élément neutre est 1 qui est le mot vide;
l'inverse de a est a^{-1}, inverse de b est b^{-1}

Modulo la relation que $1 = aa^{-1} = a^{-1}a = bb^{-1} = b^{-1}b$.

Donc $(abbba^{-1}a^{-1}b^{-1}b^{-1}b^{-1})(bbbaab^{-1}) = abb$

et $(abbba^{-1}a^{-1}b^{-1}b^{-1}b^{-1})^{-1} = bbbaab^{-1}b^{-1}b^{-1}a^{-1}$

En matrices on peut prendre $a = \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}$ et $b = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}$

On peut faire la même chose avec autant de générateurs qu'on veut.

Le groupe libre $F(a, b)$ sur $\{a, b\}$

L'ensemble est tous les "mots" en a, b, a^{-1}, b^{-1} modulo une relation;

L'opération est concatenation des mots;
l'élément neutre est 1 qui est le mot vide;
l'inverse de a est a^{-1}, inverse de b est b^{-1}

Modulo la relation que $1 = aa^{-1} = a^{-1}a = bb^{-1} = b^{-1}b$.

Donc $(abbba^{-1}a^{-1}b^{-1}b^{-1}b^{-1})(bbbaab^{-1}) = abb$
et $(abbba^{-1}a^{-1}b^{-1}b^{-1}b^{-1})^{-1} = bbbaab^{-1}b^{-1}b^{-1}a^{-1}$

En matrices on peut prendre $a = \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}$ et $b = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}$

On peut faire la même chose avec autant de générateurs qu'on veut.

Le groupe libre $F(a, b)$ sur $\{a, b\}$

L'ensemble est tous les "mots" en a, b, a^{-1}, b^{-1} modulo une relation;

L'opération est concatenation des mots;
l'élément neutre est 1 qui est le mot vide;
l'inverse de a est a^{-1}, inverse de b est b^{-1}

Modulo la relation que $1 = aa^{-1} = a^{-1}a = bb^{-1} = b^{-1}b$.

Donc $(abbba^{-1}a^{-1}b^{-1}b^{-1}b^{-1})(bbbaab^{-1}) = abb$
et $(abbba^{-1}a^{-1}b^{-1}b^{-1}b^{-1})^{-1} = bbbaab^{-1}b^{-1}b^{-1}a^{-1}$

En matrices on peut prendre $a = \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}$ et $b = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}$

On peut faire la même chose avec autant de générateurs qu'on veut.

Le groupe libre $F(a, b)$ sur $\{a, b\}$

L'ensemble est tous les "mots" en a, b, a^{-1}, b^{-1} modulo une relation;

L'opération est concatenation des mots;
l'élément neutre est 1 qui est le mot vide;
l'inverse de a est a^{-1}, inverse de b est b^{-1}

Modulo la relation que $1 = aa^{-1} = a^{-1}a = bb^{-1} = b^{-1}b$.

Donc $(abbba^{-1}a^{-1}b^{-1}b^{-1}b^{-1})(bbbaab^{-1}) = abb$
et $(abbba^{-1}a^{-1}b^{-1}b^{-1}b^{-1})^{-1} = bbbaab^{-1}b^{-1}b^{-1}a^{-1}$

En matrices on peut prendre $a = \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}$ et $b = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}$

On peut faire la même chose avec autant de générateurs qu'on veut.

Le groupe $\mathbb{Z} \times \mathbb{Z}$ est la même chose, sauf on ajoute les relations $aba^{-1}b^{-1} = 1$ qui est équivalent à $ab = ba$ et/ou $a^{-1}b = ba^{-1}$ ....

On dit que $\langle a, b \mid aba^{-1}b^{-1} \rangle$ est une **presentation finie** de $\mathbb{Z} \times \mathbb{Z}$.

On a aussi $\langle a \mid a^N = 1 \rangle$ comme presentation finie de $\frac{Z}{N\mathbb{Z}}$.

$\{1\} \cong \langle x \mid x \rangle = \langle x, y \mid x, y \rangle = \langle x, y \mid x^7, y^{10}, xy \rangle$

$SL(2, Z) = \langle s, t \mid s^4, t^6, s^2t^3 \rangle$

Les matrices $s = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$ $t = \begin{pmatrix} 0 & -1 \\ 1 & 1 \end{pmatrix}$ engendrent.

Le groupe $\mathbb{Z} \times \mathbb{Z}$ est la même chose, sauf on ajoute les relations $aba^{-1}b^{-1} = 1$ qui est équivalent à $ab = ba$ et/ou $a^{-1}b = ba^{-1}$ ....

On dit que $\langle a, b \mid aba^{-1}b^{-1} \rangle$ est une **presentation finie** de $\mathbb{Z} \times \mathbb{Z}$.
On a aussi $\langle a \mid a^N = 1 \rangle$ comme presentation finie de $\frac{Z}{N\mathbb{Z}}$.

$\{1\} \cong \langle x \mid x \rangle = \langle x, y \mid x, y \rangle = \langle x, y \mid x^7, y^{10}, xy \rangle$

$SL(2, Z) = \langle s, t \mid s^4, t^6, s^2t^3 \rangle$

Les matrices $s = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$ $t = \begin{pmatrix} 0 & -1 \\ 1 & 1 \end{pmatrix}$ engendrent.

Le groupe $\mathbb{Z} \times \mathbb{Z}$ est la même chose, sauf on ajoute les relations $aba^{-1}b^{-1} = 1$ qui est équivalent à $ab = ba$ et/ou $a^{-1}b = ba^{-1}$ ....

On dit que $\langle a, b \mid aba^{-1}b^{-1} \rangle$ est une **presentation finie** de $\mathbb{Z} \times \mathbb{Z}$.

On a aussi $\langle a \mid a^N = 1 \rangle$ comme presentation finie de $\frac{Z}{N\mathbb{Z}}$.

$\{1\} \cong \langle x \mid x \rangle = \langle x, y \mid x, y \rangle = \langle x, y \mid x^7, y^{10}, xy \rangle$

$SL(2, Z) = \langle s, t \mid s^4, t^6, s^2t^3 \rangle$

Les matrices $s = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$ $t = \begin{pmatrix} 0 & -1 \\ 1 & 1 \end{pmatrix}$ engendrent.

Le groupe $\mathbb{Z} \times \mathbb{Z}$ est la même chose, sauf on ajoute les relations $aba^{-1}b^{-1} = 1$ qui est équivalent à $ab = ba$ et/ou $a^{-1}b = ba^{-1}$ ....

On dit que $\langle a, b \mid aba^{-1}b^{-1} \rangle$ est une **presentation finie** de $\mathbb{Z} \times \mathbb{Z}$.

On a aussi $\langle a \mid a^N = 1 \rangle$ comme presentation finie de $\frac{Z}{N\mathbb{Z}}$.

$\{1\} \cong \langle x \mid x \rangle = \langle x, y \mid x, y \rangle = \langle x, y \mid x^7, y^{10}, xy \rangle$

$SL(2, Z) = \langle s, t \mid s^4, t^6, s^2t^3 \rangle$

Les matrices $s = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$ $t = \begin{pmatrix} 0 & -1 \\ 1 & 1 \end{pmatrix}$ engendrent.

Les problèmes de decision de Dehn (1911):

Problème de mot: étant donnée une présentation finie, donner un algorithme qui prend pour input un mot dans les générateurs, et répond oui ou non, selon si le mot represente l'élément trivial du groupe ou non.

Problème de conjugaison: étant donnée une présentation finie, donner un algorithme qui prend pour input deux mots dans les générateurs, et répond oui ou non, selon si les mots sont conjugués dans le groupe ou non.

Problème d'isomorphisme: Donner un algorithme, qui prend pour input deux présentations finies et répond oui ou non, selon si les groupes présentés sont isomorphes ou non.

Problème de conjugaison multiple: Étant donnée une présentation finie et deux ensembles ordonnés de k mots dans les générateurs, $\{u_1, \ldots, u_k\}$, $\{v_1, v_2, \ldots, v_k\}$ donner un algorithme qui répond oui ou non, selon s'il existe ou non un mot w tel que $wu_iw^{-1} = v_i$ pour tous les i, ET si la reponse est "oui", trouve un tel mot.

Tous les problèmes sont résolubles dans les groupes finis ou abéliens.

Les problèmes de decision de Dehn (1911):

Problème de mot: étant donnée une présentation finie, donner un algorithme qui prend pour input un mot dans les générateurs, et répond oui ou non, selon si le mot represente l'élément trivial du groupe ou non.

Problème de conjugaison: étant donnée une présentation finie, donner un algorithme qui prend pour input deux mots dans les générateurs, et répond oui ou non, selon si les mots sont conjugués dans le groupe ou non.

Problème d'isomorphisme: Donner un algorithme, qui prend pour input deux présentations finies et répond oui ou non, selon si les groupes présentés sont isomorphes ou non.

Problème de conjugaison multiple: Étant donnée une présentation finie et deux ensembles ordonnés de k mots dans les générateurs, $\{u_1, \ldots, u_k\}$, $\{v_1, v_2, \ldots, v_k\}$ donner un algorithme qui répond oui ou non, selon s'il existe ou non un mot w tel que $wu_iw^{-1} = v_i$ pour tous les i, ET si la reponse est "oui", trouve un tel mot.

Tous les problèmes sont résolubles dans les groupes finis ou abéliens.

Les problèmes de decision de Dehn (1911):

Problème de mot: étant donnée une présentation finie, donner un algorithme qui prend pour input un mot dans les générateurs, et répond oui ou non, selon si le mot represente l'élément trivial du groupe ou non.

Problème de conjugaison: étant donnée une présentation finie, donner un algorithme qui prend pour input deux mots dans les générateurs, et répond oui ou non, selon si les mots sont conjugués dans le groupe ou non.

Problème d'isomorphisme: Donner un algorithme, qui prend pour input deux présentations finies et répond oui ou non, selon si les groupes présentés sont isomorphes ou non.

Problème de conjugaison multiple: Étant donnée une présentation finie et deux ensembles ordonnés de k mots dans les générateurs, $\{u_1, \ldots, u_k\}$, $\{v_1, v_2, \ldots, v_k\}$ donner un algorithme qui répond oui ou non, selon s'il existe ou non un mot w tel que $wu_iw^{-1} = v_i$ pour tous les i, ET si la reponse est "oui", trouve un tel mot.

Tous les problèmes sont résolubles dans les groupes finis ou abéliens.

Les problèmes de decision de Dehn (1911):

Problème de mot: étant donnée une présentation finie, donner un algorithme qui prend pour input un mot dans les générateurs, et répond oui ou non, selon si le mot represente l'élément trivial du groupe ou non.

Problème de conjugaison: étant donnée une présentation finie, donner un algorithme qui prend pour input deux mots dans les générateurs, et répond oui ou non, selon si les mots sont conjugués dans le groupe ou non.

Problème d'isomorphisme: Donner un algorithme, qui prend pour input deux présentations finies et répond oui ou non, selon si les groupes présentés sont isomorphes ou non.

Problème de conjugaison multiple: Étant donnée une présentation finie et deux ensembles ordonnés de k mots dans les générateurs, $\{u_1, \ldots, u_k\}$, $\{v_1, v_2, \ldots, v_k\}$ donner un algorithme qui répond oui ou non, selon s'il existe ou non un mot w tel que $wu_iw^{-1} = v_i$ pour tous les i, ET si la reponse est "oui", trouve un tel mot.

Tous les problèmes sont résolubles dans les groupes finis ou abéliens.

Les problèmes de decision de Dehn (1911):

Problème de mot: étant donnée une présentation finie, donner un algorithme qui prend pour input un mot dans les générateurs, et répond oui ou non, selon si le mot represente l'élément trivial du groupe ou non.

Problème de conjugaison: étant donnée une présentation finie, donner un algorithme qui prend pour input deux mots dans les générateurs, et répond oui ou non, selon si les mots sont conjugués dans le groupe ou non.

Problème d'isomorphisme: Donner un algorithme, qui prend pour input deux présentations finies et répond oui ou non, selon si les groupes présentés sont isomorphes ou non.

Problème de conjugaison multiple: Étant donnée une présentation finie et deux ensembles ordonnés de k mots dans les générateurs, $\{u_1, \ldots, u_k\}$, $\{v_1, v_2, \ldots, v_k\}$ donner un algorithme qui répond oui ou non, selon s'il existe ou non un mot w tel que $wu_iw^{-1} = v_i$ pour tous les i, ET si la reponse est "oui", trouve un tel mot.

Tous les problèmes sont résolubles dans les groupes finis ou abéliens.

Les problèmes de decision de Dehn (1911):

Problème de mot: étant donnée une présentation finie, donner un algorithme qui prend pour input un mot dans les générateurs, et répond oui ou non, selon si le mot represente l'élément trivial du groupe ou non.

Problème de conjugaison: étant donnée une présentation finie, donner un algorithme qui prend pour input deux mots dans les générateurs, et répond oui ou non, selon si les mots sont conjugués dans le groupe ou non.

Problème d'isomorphisme: Donner un algorithme, qui prend pour input deux présentations finies et répond oui ou non, selon si les groupes présentés sont isomorphes ou non.

Problème de conjugaison multiple: Étant donnée une présentation finie et deux ensembles ordonnés de k mots dans les générateurs, $\{u_1, \ldots, u_k\}$, $\{v_1, v_2, \ldots, v_k\}$ donner un algorithme qui répond oui ou non, selon s'il existe ou non un mot w tel que $wu_iw^{-1} = v_i$ pour tous les i, ET si la reponse est "oui", trouve un tel mot.

Tous les problèmes sont résolubles dans les groupes finis ou abéliens.

Version 3.0 (AAG)

1. Clés publiques: une présentation finie $\langle X, R \rangle$ d'un groupe G. Alice publie une liste $X_A = \{u_1, \ldots, u_m\}$ de mots dans $F(X)$. Bob publie une liste $X_B = \{v_1, \ldots, v_n\}$ de mots dans $F(X)$.
2. Clés secrètes:
   Alice choisit un mot $\alpha(X_B)$ dans les générateurs de Bob.
   Bob choisit un mot $\beta(X_A)$ dans les générateurs de Alice.
3. Alice envoie $\alpha X_A \alpha^{-1} = \{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$ à Bob.
   Bob envoie $\beta X_B \beta^{-1} = \{\beta v_1 \beta^{-1}, \ldots, \beta v_m \beta^{-1}\}$ à Alice.
4. Bob peut maintenant recalculer son mot $\beta(X_A)$ en termes de $\{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$.
   Mais $\beta(\alpha X_A \alpha^{-1}) =_G \alpha\beta(X_A)\alpha^{-1}$.
   Bob peut aussi calculer β^{-1} et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
5. Alice peut maintenant recalculer son mot $\alpha(X_B)$ en termes de $\{\beta v_1 \beta^{-1}, \ldots, \beta v_n \beta^{-1}\}$. Mais $\alpha(\beta X_B \beta^{-1}) =_G \beta\alpha(X_B)\beta^{-1}$.
   Alice peut prendre l'inverse $\beta\alpha^{-1}\beta^{-1}$ et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
6. Clé secrète partagée : $\alpha\beta\alpha^{-1}\beta^{-1}$.

Version 3.0 (AAG)

1. Clés publiques: une présentation finie $\langle X, R \rangle$ d'un groupe G.
   Alice publie une liste $X_A = \{u_1, \ldots, u_m\}$ de mots dans $F(X)$. Bob publie une liste $X_B = \{v_1, \ldots, v_n\}$ de mots dans $F(X)$.
2. Clés secrètes:
   Alice choisit un mot $\alpha(X_B)$ dans les générateurs de Bob.
   Bob choisit un mot $\beta(X_A)$ dans les génerateurs de Alice.
3. Alice envoie $\alpha X_A \alpha^{-1} = \{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$ à Bob.
   Bob envoie $\beta X_B \beta^{-1} = \{\beta v_1 \beta^{-1}, \ldots, \beta v_m \beta^{-1}\}$ à Alice.
4. Bob peut maintenant recalculer son mot $\beta(X_A)$ en termes de $\{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$.
   Mais $\beta(\alpha X_A \alpha^{-1}) =_G \alpha\beta(X_A)\alpha^{-1}$.
   Bob peut aussi calculer β^{-1} et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
5. Alice peut maintenant recalculer son mot $\alpha(X_B)$ en termes de $\{\beta v_1 \beta^{-1}, \ldots, \beta v_n \beta^{-1}\}$. Mais $\alpha(\beta X_B \beta^{-1}) =_G \beta\alpha(X_B)\beta^{-1}$.
   Alice peut prendre l'inverse $\beta\alpha^{-1}\beta^{-1}$ et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
6. Clé secrète partagée : $\alpha\beta\alpha^{-1}\beta^{-1}$.

Version 3.0 (AAG)

1. Clés publiques: une présentation finie $\langle X, R \rangle$ d'un groupe G. Alice publie une liste $X_A = \{u_1, \ldots, u_m\}$ de mots dans $F(X)$. Bob publie une liste $X_B = \{v_1, \ldots, v_n\}$ de mots dans $F(X)$.
2. Clés secrètes:
   Alice choisit un mot $\alpha(X_B)$ dans les générateurs de Bob.
   Bob choisit un mot $\beta(X_A)$ dans les génerateurs de Alice.
3. Alice envoie $\alpha X_A \alpha^{-1} = \{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$ à Bob.
   Bob envoie $\beta X_B \beta^{-1} = \{\beta v_1 \beta^{-1}, \ldots, \beta v_m \beta^{-1}\}$ à Alice.
4. Bob peut maintenant recalculer son mot $\beta(X_A)$ en termes de $\{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$.
   Mais $\beta(\alpha X_A \alpha^{-1}) =_G \alpha\beta(X_A)\alpha^{-1}$.
   Bob peut aussi calculer β^{-1} et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
5. Alice peut maintenant recalculer son mot $\alpha(X_B)$ en termes de $\{\beta v_1 \beta^{-1}, \ldots, \beta v_n \beta^{-1}\}$. Mais $\alpha(\beta X_B \beta^{-1}) =_G \beta\alpha(X_B)\beta^{-1}$.
   Alice peut prendre l'inverse $\beta\alpha^{-1}\beta^{-1}$ et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
6. Clé secrète partagée : $\alpha\beta\alpha^{-1}\beta^{-1}$.

Version 3.0 (AAG)

1. Clés publiques: une présentation finie $\langle X, R \rangle$ d'un groupe G.
   Alice publie une liste $X_A = \{u_1, \ldots, u_m\}$ de mots dans $F(X)$. Bob publie une liste $X_B = \{v_1, \ldots, v_n\}$ de mots dans $F(X)$.
2. Clés secrètes:
   Alice choisit un mot $\alpha(X_B)$ dans les générateurs de Bob.
   Bob choisit un mot $\beta(X_A)$ dans les génerateurs de Alice.
3. Alice envoie $\alpha X_A \alpha^{-1} = \{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$ à Bob.
   Bob envoie $\beta X_B \beta^{-1} = \{\beta v_1 \beta^{-1}, \ldots, \beta v_m \beta^{-1}\}$ à Alice.
4. Bob peut maintenant recalculer son mot $\beta(X_A)$ en termes de $\{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$.
   Mais $\beta(\alpha X_A \alpha^{-1}) =_G \alpha\beta(X_A)\alpha^{-1}$.
   Bob peut aussi calculer β^{-1} et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
5. Alice peut maintenant recalculer son mot $\alpha(X_B)$ en termes de $\{\beta v_1 \beta^{-1}, \ldots, \beta v_n \beta^{-1}\}$. Mais $\alpha(\beta X_B \beta^{-1}) =_G \beta\alpha(X_B)\beta^{-1}$.
   Alice peut prendre l'inverse $\beta\alpha^{-1}\beta^{-1}$ et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
6. Clé secrète partagée : $\alpha\beta\alpha^{-1}\beta^{-1}$.

Version 3.0 (AAG)

1. Clés publiques: une présentation finie $\langle X, R \rangle$ d'un groupe G.
   Alice publie une liste $X_A = \{u_1, \ldots, u_m\}$ de mots dans $F(X)$. Bob publie une liste $X_B = \{v_1, \ldots, v_n\}$ de mots dans $F(X)$.
2. Clés secrètes:
   Alice choisit un mot $\alpha(X_B)$ dans les générateurs de Bob.
   Bob choisit un mot $\beta(X_A)$ dans les génerateurs de Alice.
3. Alice envoie $\alpha X_A \alpha^{-1} = \{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$ à Bob.
   Bob envoie $\beta X_B \beta^{-1} = \{\beta v_1 \beta^{-1}, \ldots, \beta v_m \beta^{-1}\}$ à Alice.
4. Bob peut maintenant recalculer son mot $\beta(X_A)$ en termes de $\{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$.
   Mais $\beta(\alpha X_A \alpha^{-1}) =_G \alpha\beta(X_A)\alpha^{-1}$.
   Bob peut aussi calculer β^{-1} et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
5. Alice peut maintenant recalculer son mot $\alpha(X_B)$ en termes de $\{\beta v_1 \beta^{-1}, \ldots, \beta v_n \beta^{-1}\}$. Mais $\alpha(\beta X_B \beta^{-1}) =_G \beta\alpha(X_B)\beta^{-1}$.
   Alice peut prendre l'inverse $\beta\alpha^{-1}\beta^{-1}$ et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
6. Clé secrète partagée : $\alpha\beta\alpha^{-1}\beta^{-1}$.

Version 3.0 (AAG)

1. Clés publiques: une présentation finie $\langle X, R \rangle$ d'un groupe G.
   Alice publie une liste $X_A = \{u_1, \ldots, u_m\}$ de mots dans $F(X)$. Bob publie une liste $X_B = \{v_1, \ldots, v_n\}$ de mots dans $F(X)$.
2. Clés secrètes:
   Alice choisit un mot $\alpha(X_B)$ dans les générateurs de Bob.
   Bob choisit un mot $\beta(X_A)$ dans les génerateurs de Alice.
3. Alice envoie $\alpha X_A \alpha^{-1} = \{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$ à Bob.
   Bob envoie $\beta X_B \beta^{-1} = \{\beta v_1 \beta^{-1}, \ldots, \beta v_m \beta^{-1}\}$ à Alice.
4. Bob peut maintenant recalculer son mot $\beta(X_A)$ en termes de $\{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$.
   Mais $\beta(\alpha X_A \alpha^{-1}) =_G \alpha\beta(X_A)\alpha^{-1}$.
   Bob peut aussi calculer β^{-1} et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
5. Alice peut maintenant recalculer son mot $\alpha(X_B)$ en termes de $\{\beta v_1 \beta^{-1}, \ldots, \beta v_n \beta^{-1}\}$. Mais $\alpha(\beta X_B \beta^{-1}) =_G \beta\alpha(X_B)\beta^{-1}$.
   Alice peut prendre l'inverse $\beta\alpha^{-1}\beta^{-1}$ et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
6. Clé secrète partagée : $\alpha\beta\alpha^{-1}\beta^{-1}$.

Version 3.0 (AAG)

1. Clés publiques: une présentation finie $\langle X, R \rangle$ d'un groupe G.
   Alice publie une liste $X_A = \{u_1, \ldots, u_m\}$ de mots dans $F(X)$. Bob publie une liste $X_B = \{v_1, \ldots, v_n\}$ de mots dans $F(X)$.
2. Clés secrètes:
   Alice choisit un mot $\alpha(X_B)$ dans les générateurs de Bob.
   Bob choisit un mot $\beta(X_A)$ dans les génerateurs de Alice.
3. Alice envoie $\alpha X_A \alpha^{-1} = \{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$ à Bob.
   Bob envoie $\beta X_B \beta^{-1} = \{\beta v_1 \beta^{-1}, \ldots, \beta v_m \beta^{-1}\}$ à Alice.
4. Bob peut maintenant recalculer son mot $\beta(X_A)$ en termes de $\{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$.
   Mais $\beta(\alpha X_A \alpha^{-1}) =_G \alpha\beta(X_A)\alpha^{-1}$.
   Bob peut aussi calculer β^{-1} et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
5. Alice peut maintenant recalculer son mot $\alpha(X_B)$ en termes de $\{\beta v_1 \beta^{-1}, \ldots, \beta v_n \beta^{-1}\}$. Mais $\alpha(\beta X_B \beta^{-1}) =_G \beta\alpha(X_B)\beta^{-1}$.
   Alice peut prendre l'inverse $\beta\alpha^{-1}\beta^{-1}$ et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
6. Clé secrète partagée : $\alpha\beta\alpha^{-1}\beta^{-1}$.

1. Clés publiques: une présentation finie $\langle X, R\rangle$ d'un groupe G. Alice publie une liste $X_A = \{u_1, \ldots, u_m\}$ de mots dans $F(X)$. Bob publie une liste $X_B = \{v_1, \ldots, v_n\}$ de mots dans $F(X)$.
2. Clés secrètes:
   Alice choisit un mot $\alpha(X_B)$ dans les générateurs de Bob.
   Bob choisit un mot $\beta(X_A)$ dans les génerateurs de Alice.
3. Alice envoie $\alpha X_A \alpha^{-1} = \{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$ à Bob.
   Bob envoie $\beta X_B \beta^{-1} = \{\beta v_1 \beta^{-1}, \ldots, \beta v_m \beta^{-1}\}$ à Alice.
4. Bob peut maintenant recalculer son mot $\beta(X_A)$ en termes de $\{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$.
   Mais $\beta(\alpha X_A \alpha^{-1}) =_G \alpha\beta(X_A)\alpha^{-1}$.
   Bob peut aussi calculer β^{-1} et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
5. Alice peut maintenant recalculer son mot $\alpha(X_B)$ en termes de $\{\beta v_1 \beta^{-1}, \ldots, \beta v_n \beta^{-1}\}$. Mais $\alpha(\beta X_B \beta^{-1}) =_G \beta\alpha(X_B)\beta^{-1}$.
   Alice peut prendre l'inverse $\beta\alpha^{-1}\beta^{-1}$ et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
6. Clé secrète partagée : $\alpha\beta\alpha^{-1}\beta^{-1}$.

Version 3.0 (AAG)

1. Clés publiques: une présentation finie $\langle X, R\rangle$ d'un groupe G.
   Alice publie une liste $X_A = \{u_1, \ldots, u_m\}$ de mots dans $F(X)$. Bob publie une liste $X_B = \{v_1, \ldots, v_n\}$ de mots dans $F(X)$.
2. Clés secrètes:
   Alice choisit un mot $\alpha(X_B)$ dans les générateurs de Bob.
   Bob choisit un mot $\beta(X_A)$ dans les génerateurs de Alice.
3. Alice envoie $\alpha X_A \alpha^{-1} = \{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$ à Bob.
   Bob envoie $\beta X_B \beta^{-1} = \{\beta v_1 \beta^{-1}, \ldots, \beta v_m \beta^{-1}\}$ à Alice.
4. Bob peut maintenant recalculer son mot $\beta(X_A)$ en termes de $\{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$.
   Mais $\beta(\alpha X_A \alpha^{-1}) =_G \alpha\beta(X_A)\alpha^{-1}$.
   Bob peut aussi calculer β^{-1} et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
5. Alice peut maintenant recalculer son mot $\alpha(X_B)$ en termes de $\{\beta v_1 \beta^{-1}, \ldots, \beta v_n \beta^{-1}\}$. Mais $\alpha(\beta X_B \beta^{-1}) =_G \beta\alpha(X_B)\beta^{-1}$.
   Alice peut prendre l'inverse $\beta\alpha^{-1}\beta^{-1}$ et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
6. Clé secrète partagée : $\alpha\beta\alpha^{-1}\beta^{-1}$.

1. Clés publiques: une présentation finie $\langle X, R\rangle$ d'un groupe G.
   Alice publie une liste $X_A = \{u_1, \ldots, u_m\}$ de mots dans $F(X)$. Bob publie une liste $X_B = \{v_1, \ldots, v_n\}$ de mots dans $F(X)$.
2. Clés secrètes:
   Alice choisit un mot $\alpha(X_B)$ dans les générateurs de Bob.
   Bob choisit un mot $\beta(X_A)$ dans les génerateurs de Alice.
3. Alice envoie $\alpha X_A \alpha^{-1} = \{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$ à Bob.
   Bob envoie $\beta X_B \beta^{-1} = \{\beta v_1 \beta^{-1}, \ldots, \beta v_m \beta^{-1}\}$ à Alice.
4. Bob peut maintenant recalculer son mot $\beta(X_A)$ en termes de $\{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$.
   Mais $\beta(\alpha X_A \alpha^{-1}) =_G \alpha\beta(X_A)\alpha^{-1}$.
   Bob peut aussi calculer β^{-1} et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
5. Alice peut maintenant recalculer son mot $\alpha(X_B)$ en termes de $\{\beta v_1 \beta^{-1}, \ldots, \beta v_n \beta^{-1}\}$. Mais $\alpha(\beta X_B \beta^{-1}) =_G \beta\alpha(X_B)\beta^{-1}$.
   Alice peut prendre l'inverse $\beta\alpha^{-1}\beta^{-1}$ et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
6. Clé secrète partagée : $\alpha\beta\alpha^{-1}\beta^{-1}$.

1. Clés publiques: une présentation finie $\langle X, R\rangle$ d'un groupe G. Alice publie une liste $X_A = \{u_1, \ldots, u_m\}$ de mots dans $F(X)$. Bob publie une liste $X_B = \{v_1, \ldots, v_n\}$ de mots dans $F(X)$.
2. Clés secrètes:
   Alice choisit un mot $\alpha(X_B)$ dans les générateurs de Bob.
   Bob choisit un mot $\beta(X_A)$ dans les génerateurs de Alice.
3. Alice envoie $\alpha X_A \alpha^{-1} = \{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$ à Bob.
   Bob envoie $\beta X_B \beta^{-1} = \{\beta v_1 \beta^{-1}, \ldots, \beta v_m \beta^{-1}\}$ à Alice.
4. Bob peut maintenant recalculer son mot $\beta(X_A)$ en termes de $\{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$.
   Mais $\beta(\alpha X_A \alpha^{-1}) =_G \alpha\beta(X_A)\alpha^{-1}$.
   Bob peut aussi calculer β^{-1} et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
5. Alice peut maintenant recalculer son mot $\alpha(X_B)$ en termes de $\{\beta v_1 \beta^{-1}, \ldots, \beta v_n \beta^{-1}\}$. Mais $\alpha(\beta X_B \beta^{-1}) =_G \beta\alpha(X_B)\beta^{-1}$.
   Alice peut prendre l'inverse $\beta\alpha^{-1}\beta^{-1}$ et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
6. Clé secrète partagée : $\alpha\beta\alpha^{-1}\beta^{-1}$.

1. Clés publiques: une présentation finie $\langle X, R\rangle$ d'un groupe G.
   Alice publie une liste $X_A = \{u_1, \ldots, u_m\}$ de mots dans $F(X)$. Bob publie une liste $X_B = \{v_1, \ldots, v_n\}$ de mots dans $F(X)$.
2. Clés secrètes:
   Alice choisit un mot $\alpha(X_B)$ dans les générateurs de Bob.
   Bob choisit un mot $\beta(X_A)$ dans les génerateurs de Alice.
3. Alice envoie $\alpha X_A \alpha^{-1} = \{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$ à Bob.
   Bob envoie $\beta X_B \beta^{-1} = \{\beta v_1 \beta^{-1}, \ldots, \beta v_m \beta^{-1}\}$ à Alice.
4. Bob peut maintenant recalculer son mot $\beta(X_A)$ en termes de $\{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$.
   Mais $\beta(\alpha X_A \alpha^{-1}) =_G \alpha\beta(X_A)\alpha^{-1}$.
   Bob peut aussi calculer β^{-1} et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
5. Alice peut maintenant recalculer son mot $\alpha(X_B)$ en termes de $\{\beta v_1 \beta^{-1}, \ldots, \beta v_n \beta^{-1}\}$. Mais $\alpha(\beta X_B \beta^{-1}) =_G \beta\alpha(X_B)\beta^{-1}$.
   Alice peut prendre l'inverse $\beta\alpha^{-1}\beta^{-1}$ et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
6. Clé secrète partagée : $\alpha\beta\alpha^{-1}\beta^{-1}$.

1. Clés publiques: une présentation finie $\langle X, R\rangle$ d'un groupe G.
   Alice publie une liste $X_A = \{u_1, \ldots, u_m\}$ de mots dans $F(X)$. Bob publie une liste $X_B = \{v_1, \ldots, v_n\}$ de mots dans $F(X)$.
2. Clés secrètes:
   Alice choisit un mot $\alpha(X_B)$ dans les générateurs de Bob.
   Bob choisit un mot $\beta(X_A)$ dans les génerateurs de Alice.
3. Alice envoie $\alpha X_A \alpha^{-1} = \{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$ à Bob.
   Bob envoie $\beta X_B \beta^{-1} = \{\beta v_1 \beta^{-1}, \ldots, \beta v_m \beta^{-1}\}$ à Alice.
4. Bob peut maintenant recalculer son mot $\beta(X_A)$ en termes de $\{\alpha u_1 \alpha^{-1}, \ldots, \alpha u_m \alpha^{-1}\}$.
   Mais $\beta(\alpha X_A \alpha^{-1}) =_G \alpha\beta(X_A)\alpha^{-1}$.
   Bob peut aussi calculer β^{-1} et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
5. Alice peut maintenant recalculer son mot $\alpha(X_B)$ en termes de $\{\beta v_1 \beta^{-1}, \ldots, \beta v_n \beta^{-1}\}$. Mais $\alpha(\beta X_B \beta^{-1}) =_G \beta\alpha(X_B)\beta^{-1}$.
   Alice peut prendre l'inverse $\beta\alpha^{-1}\beta^{-1}$ et donc calculer $\alpha\beta\alpha^{-1}\beta^{-1}$.
6. Clé secrète partagée : $\alpha\beta\alpha^{-1}\beta^{-1}$.

Avec cette clé privée, une version simplifié du codage est :

Soit T le message. . . un mot longue en $\{0,1\}^M$.

Supposons connue la function $H : G \to \{0,1\}^M$ ("hashing function")

A et B connaissent donc $H(\alpha\beta\alpha^{-1}\beta^{-1})$.

Pour envoyer T :

$T' = H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus T$ où $\oplus$ est la somme mod 2 de chaque chiffre.

Pour décoder il suffit de refaire

$$H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus T' = H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus T = T$$

SECURITÉ = choisir un groupe avec problème de conjugaison insoluble!!

Le groupe mode est le groupe des tresses B_n

Il reste a montrer que la solution du problème de conjugaison est difficile à résoudre dans B_n.

Avec cette clé privée, une version simplifié du codage est :
Soit T le message. . . un mot longue en $\{0,1\}^M$.
Supposons connue la function $H : G \to \{0,1\}^M$ ("hashing function")
A et B connaissent donc $H(\alpha\beta\alpha^{-1}\beta^{-1})$.
Pour envoyer T :
$T' = H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus T$ où $\oplus$ est la somme mod 2 de chaque chiffre.
Pour décoder il suffit de refaire

$$H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus T' = H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus T = T$$

SECURITÉ = choisir un groupe avec problème de conjugaison insoluble!!
Le groupe mode est le groupe des tresses B_n
Il reste a montrer que la solution du problème de conjugaison est difficile à résoudre dans B_n.

Avec cette clé privée, une version simplifié du codage est :
Soit T le message. . . un mot longue en $\{0,1\}^M$.
Supposons connue la function $H : G \to \{0,1\}^M$ ("hashing function")
A et B connaissent donc $H(\alpha\beta\alpha^{-1}\beta^{-1})$.
Pour envoyer T :
$T' = H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus T$ où $\oplus$ est la somme mod 2 de chaque chiffre.
Pour décoder il suffit de refaire

$$H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus T' = H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus T = T$$

SECURITÉ = choisir un groupe avec problème de conjugaison insoluble!!
Le groupe mode est le groupe des tresses B_n
Il reste a montrer que la solution du problème de conjugaison est difficile à résoudre dans B_n.

Avec cette clé privée, une version simplifié du codage est :
Soit T le message. . . un mot longue en $\{0,1\}^M$.
Supposons connue la function $H : G \to \{0,1\}^M$ ("hashing function")
A et B connaissent donc $H(\alpha\beta\alpha^{-1}\beta^{-1})$.
Pour envoyer T :
$T' = H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus T$ où $\oplus$ est la somme mod 2 de chaque chiffre.
Pour décoder il suffit de refaire

$$H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus T' = H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus T = T$$

SECURITÉ = choisir un groupe avec problème de conjugaison insoluble!!
Le groupe mode est le groupe des tresses B_n
Il reste a montrer que la solution du problème de conjugaison est difficile à résoudre dans B_n.

Avec cette clé privée, une version simplifié du codage est :
Soit T le message. . . un mot longue en $\{0,1\}^M$.
Supposons connue la function $H : G \to \{0,1\}^M$ ("hashing function")
A et B connaissent donc $H(\alpha\beta\alpha^{-1}\beta^{-1})$.
Pour envoyer T :
$T' = H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus T$ où $\oplus$ est la somme mod 2 de chaque chiffre.
Pour décoder il suffit de refaire

$$H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus T' = H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus T = T$$

SECURITÉ = choisir un groupe avec problème de conjugaison insoluble!!
Le groupe mode est le groupe des tresses B_n
Il reste a montrer que la solution du problème de conjugaison est difficile à résoudre dans B_n.

Avec cette clé privée, une version simplifié du codage est :
Soit T le message. . . un mot longue en $\{0,1\}^M$.
Supposons connue la function $H : G \to \{0,1\}^M$ ("hashing function")
A et B connaissent donc $H(\alpha\beta\alpha^{-1}\beta^{-1})$.
Pour envoyer T :
$T' = H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus T$ où $\oplus$ est la somme mod 2 de chaque chiffre.
Pour décoder il suffit de refaire

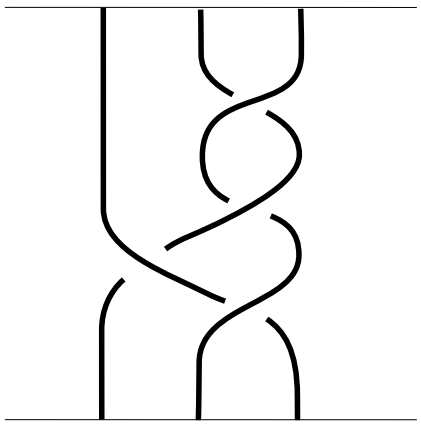
$$H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus T' = H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus H(\alpha\beta\alpha^{-1}\beta^{-1}) \oplus T = T$$

SECURITÉ = choisir un groupe avec problème de conjugaison insoluble!!
Le groupe mode est le groupe des tresses B_n
Il reste a montrer que la solution du problème de conjugaison est difficile à résoudre dans B_n.

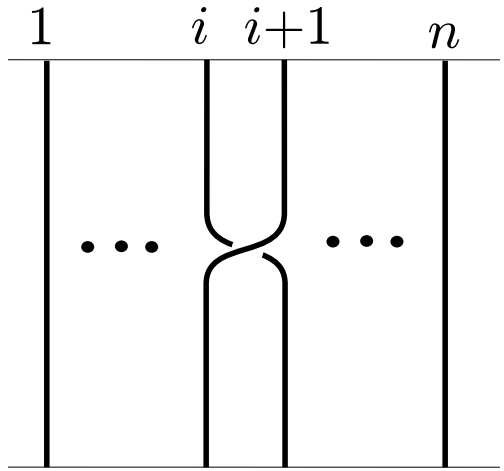
Le groupe de tresses B_q:
$\langle x_1, x_2, \ldots, x_q \mid [x_i, x_j] = 1 \text{ si } \ |i-j| > 1, x_i x_{i+1} x_i = x_{i+1} x_i x_{i+1} \rangle$

Chaque élément s'écrit $\Delta^k b$ avec b positif (sans puissance -1) et

$$\Delta = x_1(x_2x_1)(x_3x_2x_1)\ldots(x_qx_{q-1}\ldots x_1)$$



$x_2^2 x_1^{-1} x_2$



le générateur x_i