LOGIQUE MATHEMATIQUE. — Machines de Turing réversibles.

Récursif insolubilité en $n \in \mathbb{N}$ de l’équation $u = \theta^nu$, où $\theta$ est un « isomorphisme de codes ». Note (*) de M. Yves Lecerf, présentée par M. André Lichnerowicz.

On définit des « machines de Turing réversibles » et des « isomorphismes de codes $\theta$ ». Leurs propriétés permettent de prouver que l’équation en $n \in \mathbb{N}$, $u = \theta^nu$ est récursivement insoluble. Une seconde Note appliquera ceci à la démonstration d’une conjecture de Schützenberger rattachant le problème des correspondances de Post à des problèmes de diagonalisation d’homomorphismes de monoïdes libres.

1. ISOMORPHISMES DE CODES, ÉPIMORPHISMES DE CODES. — a. Une conjecture de Schützenberger. — Soient deux monoïdes libres non triviaux $A^\dagger$ et $S^\dagger$, soient deux homomorphismes $\varphi$ et $\psi$ de $A^\dagger$ dans $S^\dagger$, et soit le problème de la recherche de solutions non triviales $x \in A^\dagger$ pour l’équation de diagonalisation $\varphi x = \psi x$. Un résultat de Post (6) est que cette équation est récursivement insoluble dans le cas de $\varphi$ et $\psi$ homomorphismes quelconques. Elle l’est également lorsqu’on astreint $\varphi$ à être un monomorphisme ; Chomsky et Schützenberger ont fait observer, en effet (1), que ce cas peut se ramener au Tag-problème de Post (5) lui-même récursivement insoluble d’après un résultat de Minsky (3). Schützenberger a conjecturé que l’équation $\varphi x = \psi x$ reste encore récursivement insoluble quand à la fois $\varphi$ et $\psi$ sont deux monomorphismes.

b. Isomorphismes de codes. — Au lieu de $\varphi x = \psi x$, il revient au même de considérer l’équation $w = \theta w$, où $\theta = \psi^{-1} \varphi$ (ce qui est une notation abrégée pour dire que $\theta$ est une bijection de $\varphi A^\dagger$ dans $\psi A^\dagger$ définie par $\theta w = \psi x$ pour $w = \varphi x$). Par facilité de langage, on appellera « isomorphismes de codes » les applications telles que $\theta$. Ce terme rappelle que $\theta(w_1 w_2) = \theta w_1 \theta w_2$ et aussi que, $A = \{ a_i \}_{i \in I}$ désignant l’alphabet (les générateurs) de $A^\dagger$, $\{ \varphi a_i \}_{i \in I}$ et $\{ \psi a_i \}_{i \in I}$ sont appelés des « codes » sur $S^\dagger$, car, pour $y$ quelconque de $S^\dagger$, il existe au plus un ensemble d’indices $\{ i_1, i_2, \ldots, i_p \}$ tels que $y = \varphi a_{i_1} \varphi a_{i_2} \ldots \varphi a_{i_p}$, et de même pour $\psi$. En fait, c’est surtout à l’étude des isomorphismes de code que seront consacrées la présente Note et la suivante.

c. Définitions d’« isomorphismes de codes » particuliers par donnée de relations. — $e_A$ et $e_S$ désignant les éléments neutres respectivement de $A^\dagger$ et de $S^\dagger$, il va de soi implicitement pour tout $\theta$ que l’on a $e_S = \theta e_A$, avec $\varphi e_A = \psi e_A = e_S$ (d’ou, d’ailleurs, une solution triviale pour $w = \theta w$ et pour $w = \theta^nu$, avec $n \in \mathbb{N}$). Ceci étant, chaque isomorphisme de codes particulier pourra être défini par la donnée d’un ensemble de relations du type $\{ m_{i\varphi} \to m_{i\psi} \}_{i \in I}$, à condition que $\{ n_{i\varphi} \}_{i \in I}$ et $\{ m_{i\psi} \}_{i \in I}$ soient des « codes » et que la correspondance soit bijective. En effet, $A^\dagger$ est implicitement défini par $I$, $S^\dagger$ l’est par les symboles utilisés pour noter les $m_{i\varphi}$ et $m_{i\psi}$ ; et l’on peut interpréter les relations comme des correspondances $\{ \varphi a_i \to \psi a_i \}_{i \in I}$.
d. Vérification qu’un ensemble de mots donnés est un code. — Plus loin, on invoquera souvent la propriété suivante : Si C et $K_d$ désignent respectivement un code et un préfixe-code droit sur $S^1$, et si $\alpha$ est un symbole (générateur de $S^1$) ne figurant pas dans C ni dans $K_d$, alors, l’ensemble $C \cup \alpha K_d$ est un code. De même, en remplaçant $K_d$ par un préfixe-code gauche $K_g$, l’ensemble $C \cup K_g \alpha$ est un code. Rappelons que tout préfixe-code droit $K_d$ est par définition (4) tel que, si $m_i, m_j \in K_d$ et si, avec $y \in S^1$, on a $m_i = m_j y$, alors, $y = e_S$ (tandis que pour les préfixe-codes gauches, c’est $m_i = y m_j$ qui impose $y = e_S$).

e. Épimorphismes de codes. — On parlera d’« épimorphisme de codes » $\tau$ dans le cas de relations $\{m_{i,\rho} \rightarrow m_{i,\psi}\}_{i \in I}$, où $\{m_{i,\rho}\}_{i \in I}$ est bien un code $C_{\rho}$, mais où $\{m_{i,\psi}\}_{i \in I}$ est astreint seulement à ne pas contenir d’autres mots que ceux d’un code $C_{\psi}$.

2. Machines de Turing réversibles. — Soit une machine de Turing MT dont $\{\varepsilon_p\}_{p \in P}$ et $\{\sigma_q\}_{q \in Q}$ sont les ensembles d’états et de symboles, et $\{\delta_r\}_{r \in R}$ les déplacements de ruban, qui peuvent être $\pm 1$ ou $0$. On peut définir MT par un ensemble de quintuples

$$\chi_{MT} = \{\varepsilon_{p_1(i)}; \sigma_{q_1(i)}; \varepsilon_{p_2(i)}; \sigma_{q_2(i)}; \delta_{r(i)}\}_{i \in I},$$

où les indices $p_1, p_2, q_1, q_2, r$ sont fonctions de l’indice $i$. A chacun de ces quintuples, décidons d’associer un « quintuple image inverse » $(\varepsilon^*_p(i); \sigma^*_q(i); \varepsilon^*_p(i); \sigma^*_q(i); -\delta_{r(i)})$. L’ensemble de ceux-ci ne constituera généralement pas une machine de Turing; mais lorsqu’il en sera ainsi, on dira que MT est « réversible », et l’on donnera à la nouvelle machine le nom d’image inverse MT* de MT. Les $\varepsilon^*_p$ seront dits images des $\varepsilon_p$. La substitution d’un $\varepsilon^*_p$ à un $\varepsilon_p$ dans une configuration instantannée $U_k$ sera dite passage à $U_k$ à sa configuration image $U^*_k$. Les suites de configurations de MT* sont images de celles de MT, mais MT* les parcourt dans l’ordre inverse. Considérons maintenant la machine R (MT), dont l’ensemble de quintuples est

$$\chi_{R(MT)} = \chi_{MT} \cup \chi^*_{MT} \cup \{(\varepsilon_p; \sigma_q); \varepsilon^*_p; \sigma^*_q; 0\};$$

où $(\varepsilon_p; \sigma_q); \varepsilon^*_p$ désigne tout couple pour lequel MT stoppe. Si l’on fait partir MT et R (MT) d’une même configuration instantannée $U_0$, elles passent par les mêmes configurations tant que MT ne stoppe pas (donc éventuellement indéfiniment). Lorsque MT stoppe, R (MT) continue, parcourant en ordre inverse les configurations images des configurations parcourues, et passe par l’image de la configuration initiale. R (MT) sera dite couplage de MT avec son image inverse.

3. Représentation de machines de Turing par des épimorphismes (ou des isomorphismes) de codes. — Revenons au cas où MT est quelconque. À chacun de ses quintuples comportant le mouvement $+1$, soit par exemple $(\varepsilon_g, \sigma_h, \varepsilon_j, \sigma_k, 1)$, on associe trois relations, à savoir ici : $\{\alpha_g \sigma_h \rightarrow \sigma_j \alpha_j; \omega_g \sigma_h \rightarrow \sigma_k \alpha_j; \sigma_h \beta_g \rightarrow \sigma_k \alpha_j\}$. Aux $(\varepsilon_g, \sigma_h, \varepsilon_j, \sigma_k, 0)$, on associe : $\{\alpha_g \sigma_h \rightarrow \omega_j \sigma_k; \omega_g \sigma_h \rightarrow \omega_j \sigma_k; \sigma_h \beta_g \rightarrow \omega_j \sigma_k\}$. Aux $(\varepsilon_g, \sigma_h, \varepsilon_j, \sigma_k, -1)$, on associe $\{\alpha_g \sigma_h \rightarrow \beta_j \sigma_k; \omega_g \sigma_h \rightarrow \beta_j \sigma_k; \sigma_h \beta_g \rightarrow \beta_j \sigma_k\}$. Enfin, à tout symbole $\sigma_q$ de MT, on associe $\sigma_q \rightarrow \sigma_q$. On peut vérifier, par le procédé donné au paragraphe 1, d, que l’ensemble de toutes ces relations définit un épimorphisme de codes. Soit $\tau_{\text{max}}$ celui-ci, $\tau_{\text{max}}$ est une représentation de MT, car il en définit l’alphabet.
et les quintuples. On peut, d’autre part, trouver pour les configurations instantanées de MT des notations telles que pour tout couple de configurations successives \( u_i, u_{i+1} \) on ait \( u_{i+1} = \tau_{\max} u_i \). Pour cela, une configuration se composera d’une suite de symboles \( \sigma \) (le mot du ruban) dans laquelle on intercalerà une des lettres \( \alpha, \omega \) ou \( \beta \), avec un indice \( p \) égal à celui de l’état \( \varepsilon_p \) de la machine, et de façon à indiquer, non seulement la position \( \pi_1 \) du prochain symbole à lire, mais encore celle \( \pi_2 \) du symbole précédemment écrit (avec convention particulière pour la configuration initiale). Un \( \alpha_p \) signifie que \( \pi_1 \) est le premier symbole à sa droite, \( \pi_2 \) le premier à sa gauche. Un \( \beta_p \), l’inverse. Un \( \omega_p \), signifie que \( \pi_1 \) et \( \pi_2 \), confondus, sont tous deux le premier symbole à droite de ce \( \omega_p \). On a bien alors \( u_{i+1} = \tau_{\max} u_i \). Si certains états \( \varepsilon_p \) ne peuvent apparaître que sous deux seulement, ou une seule, des formes \( \alpha_p, \omega_p, \beta_p \), et soit \( \tau_{\min} \) obtenu en retranchant de \( \tau_{\max} \) toutes les relations contenant des formes qui n’apparaissent jamais, alors, \( \tau_{\min} \) est encore tel que \( u_{i+1} = \tau_{\min} u_i \). Si \( \tau_{\min} \) est un isomorphisme de codes, MT est réversible.

4. SIMULATION DE MT QUELCONQUE SUR MT* RÉVERSIBLE. APPLICATION AUX ISOMORPHISMES DE CODES. — a. Propriété. — On peut simuler une machine de Turing MT quelconque (de configurations \( u_i \)) sur une machine de Turing réversible MT\( \rho \) (de configurations \( u_{i,j} \)) de façon que : 1° quand MT passe de \( u_i \) à \( u_{i+1} \), MT\( \rho \) passe de \( u_{i,j} \) à \( u_{i+1,j,0} \) par l’intermédiaire d’un nombre fini de configurations \( u_{i,j}; u_{i,j}; \ldots \); 2° on passe d’un \( u_i \) au suivant par un épimorphisme de codes \( \tau \); 3° si les configurations initiales sont \( v_0 \) pour MT et \( u_{0,j} \) pour MT\( \rho \), avec \( u_{0,j} = \lambda \nu_0 \mu \nu \), alors pour tout \( i \), on a \( u_{i,j} = \lambda \nu_0 \mu \nu_\tau, \) où \( u_i \) est un mot, et où \( \lambda, \mu, \nu \) sont trois symboles qui n’apparaissent ni dans \( v_i \) ni dans \( u_i \), si bien que \( u_{i,j} \) connu donne \( v_i \) et \( u_i \); 4° il y a des symboles \( \tau_k \) dont chacun représente une relation de \( \tau \) autre que d’identité; un symbole blanc \( b \); et pour tout \( i \) on a \( w_i = b^2 \tau_k r \tau_k \ldots r \tau_k b \), où \( r \tau_k \) est la relation intervenue dans \( v_\tau = \tau v_\tau - 1 \). Ainsi, \( w_i \) représente l’histoire du calcul de MT jusqu’au temps \( \tau \); 5° MT\( \rho \) stoppe sur les \( u_{j,0} \) correspondant à des stop de MT, et eux seulement; 6° la machine R(MT\( \rho \)), coupable de MT\( \rho \) avec son image inverse, partie de \( u_{0,j} = \lambda \nu_0 \mu \nu \) passe par la configuration image \( \lambda \nu_0 \mu \nu \) si et seulement si MT, partie de \( v_0 \), stoppe; 7° il existe pour R(MT\( \rho \)) certaines configurations instantanées \( u_{\#} \) telles que, partant de \( \lambda \nu_0 \mu \nu \), R(MT\( \rho \)) ne peut atteindre ces configurations autrement qu’en passant par \( \lambda \nu_0 \mu \nu \) (c’est-à-dire si MT, partie de \( v_0 \), stoppe). On peut ainsi faire en sorte que le retour de R(MT\( \rho \)) à \( \lambda \nu_0 \mu \nu \) (ou bien, le passage par des \( u_{\#} \) encadrés par \( \lambda \nu \nu \) à lieu du \( \lambda \nu \nu \)) soit conditionné par un stop de MT.

Preuve. — On montrera comment passer de \( \tau \), supposé donné par un ensemble de relations \( \{ I_{k_{\#}} \} \) à l’ensemble des relations \( \{ I_{k_{\#}, \#} \} \) définissant \( \tau \) et MT\( \rho \), limitons-nous à donner le principe de cette construction, en montrant comment simuler une instruction \( I_{k_{\#}} \) de la forme \( \alpha_{\#} \rightarrow \alpha_{\#} \). À ce devoir, on associera : une instruction \( \alpha_{\#} \rightarrow \alpha_{\#} \), \( \alpha_{\#} \rightarrow \alpha_{\#} \), \( \alpha_{\#} \rightarrow \alpha_{\#} \), où le symbole \( \alpha_{\#} \) marque la place où l’on doit modifier \( u_\# \), et la nature de la modification; des instructions permettant de conduire le contrôle à gauche de \( \nu \) pour un état \( \varepsilon_\# \); une instruction \( b \varepsilon_\# \rightarrow \varepsilon_\# b \), où \( r_{\#} \) représente \( I_{k_{\#}} \), complétant \( u_\# \); des instructions de service déplaçant \( \nu \) et, éventuellement aussi \( \lambda, \mu \) et \( u_\# \) tout entier, pour rétablir les blancs nécessaires dans \( u_{j,0} \) puis reportant le
contrôle en \( \sigma_{f,g,\varepsilon} \) avec un état \( \varepsilon_\sigma \); une instruction \( \sigma_{f,g,\varepsilon} \varepsilon_\sigma \rightarrow \sigma_{f,\alpha_0} \) qui complète \( v_i \) dans \( u_{i,0} \).

b. Théorème 1. — Le problème du stop pour une machine de Turing réversible générale est indécidable. Il en est de même du problème du retour à la configuration initiale, et de celui du passage par une configuration déterminée autre que la configuration initiale.

c. Théorème 2. — L’équation \( w = \theta^n w \), où \( \theta \) est un isomorphisme de codes, avec \( n \in \mathbb{N} \) est récursivement insoluble en \( n \) pour \( w, \theta \) donnés quelconques. L’équation \( w_1 = \theta^n w_2 \), avec \( w_1 \neq w_2 \), est aussi récursivement insoluble en \( n \).

(*) Séance du 21 octobre 1963.

(Euratom, 51, rue Belliard, Bruxelles.)