GAMES AND ALGORITHMS WITH HOOK STRUCTURE

NORIAKI KAWANAKA

Abstract. In this paper, a maze-like branching structure is called a game or an algorithm. Branching structures are common in everyday life, in the natural world, and in mathematics; they are almost everywhere, and are extremely diverse, so that no single theory could provide a satisfactory description of them. However, if we restrict our attention to a typical type in some sense, then a mathematical treatment might be possible.

From such a viewpoint, two extreme cases look interesting; branching structures which behave chaotically on the one hand, and those which are "solvable" on the other. In this paper, we introduce a class of elementary examples, called plain algorithms, belonging to the latter, and begin to explore their game-theoretical and algorithm-theoretical properties.

1. Abstract algorithms

This section is devoted to introduce the general terminology on algorithms and games needed in later sections. Explicit examples are given in Sections 8-11. Let P be a set, and $\varphi\colon P \longrightarrow 2^P$ a map from P to the set 2^P of subsets of P. Such a pair (P,φ) will be called an (*abstract*) *algorithm*[1)]. The set P and the map φ are called the *state space* and the *branching map*, respectively. An element of P is called a *point* (or a *state*) and represents a state of the algorithm. For a state $p \in P$, $\varphi(p)$ represents the set of followers of p. Although, in a practical algorithm, such a set is usually a singleton or empty, we prefer to consider a more general case[2)]. We often write $p \to q$ (or $q \leftarrow p$) for $q \in \varphi(p)$, and call an ordered pair $p \to q$ an *arrow* with *origin* p. If $\varphi(p) = \emptyset$, there exists no follower of p; this means that the algorithm terminates at p. Such a point $p \in P$ is called an *end* of the algorithm (P,φ). An important property of an actual algorithm is that it terminates after a finite number of steps for any initial data. We say that an algorithm (P,φ) is *finite* if, for any $p \in P$, a sequence

$$p = p_0 \to p_1 \to p_2 \to \cdots \to p_i \to p_{i+1} \to \cdots \tag{1.1}$$

of points of P is always finite, namely we eventually encounter an end p_k for some k. A sequence (1.1) is called a *path* with *origin* p. We say that (P,φ) is *finitely branching*, if, for any $p \in P$, $\varphi(p)$ is a finite set. An arrow $p \to q$ is *irreducible*, if there exists no path

$$p = p_0 \to p_1 \to p_2 \to \cdots \to p_l = q \tag{1.2}$$

of length $l \geq 2$. If each arrow appearing in a path (1.1) is irreducible, we say that the path (1.1) is *irreducibly decomposed*. If (P,φ) is finite and finitely branching, then, for an arbitrary arrow $p \to q$, there exists an irreducibly decomposed path (1.2), which is called an *irreducible decomposition* of the arrow $p \to q$. Although

1991 *Mathematics Subject Classification.* Primary 05E10, 17B22, 20C30, 20F55, 91A46; Secondary 05A17, 05C57, 05A19, 20C30, 60C05.

a finite algorithm may be considered as (a model of) an algorithm that terminates after a finite number of steps, it admits other interpretations. For example, a finite algorithm (P, φ) equipped with a probability measure on each $\varphi(p) \neq \emptyset$ $(p \in P)$ may be considered as an algorithm in which a next state of $p \in P$ is selected from the set $\varphi(p)$ according to the probability measure on it. An algorithm of this type is called a *probabilistic algorithm*. A finite algorithm may also be considered as a model of a 2-*person game*[3)] in the following way. Each $p \in P$ represents a position of a game. At a position p, $\varphi(p)$ is the set of possible moves for the next player. A given position p_0 may be considered as the opening position. The *first player* selects a position $p_1 \in \varphi(p_0)$, and the *second player* selects a position $p_2 \in \varphi(p_1), \ldots$; the two players select positions alternatively. The game ends when one of the players selects an end p_n. This eventually occurs by the finiteness of (P, φ). The next player (namely, the first or the second player according as n is even or odd) is the *looser* and the other is the *winner*.

Let us introduce some notation. Given a map $\varphi\colon P \longrightarrow 2^P$, we define maps φ^k $(k = -1, 0, 1, 2, \ldots)$ and $\varphi^{\pm 1}$ from P to 2^P by putting, for $p \in P$,

$$\varphi^{-1}(p) = \{q \in P \mid q \to p\}, \quad \varphi^{\pm 1}(p) = \varphi(p) \cup \varphi^{-1}(p),$$

$$\varphi^0(p) = \{p\}, \quad \varphi^k(p) = \varphi(\varphi^{k-1}(p)) \ (k = 1, 2, 3, \ldots).$$

For example, the algorithm (P, φ^{-1}) is obtained from (P, φ) by reversing all arrows.

Let (P, φ) be an algorithm. Let Q be a subset of P. Defining $\varphi_Q\colon Q \longrightarrow 2^Q$ by

$$\varphi_Q(q) = \varphi(q) \cap Q,$$

we get an algorithm (Q, φ_Q). We call (Q, φ_Q) (or, simply Q) a *subalgorithm* of (P, φ). Thus, a subset of the state space P is always considered as the state space of a subalgorithm. In the particular case when Q is closed under φ, namely when $\varphi(q) \subset Q$ holds for any $q \in Q$, we call Q a *full* subalgorithm. The minimum full subalgorithm containing $p \in P$ is denoted by $\langle p \rangle_\varphi$, which is also defined by

$$\langle p \rangle_\varphi = \bigcup_{k=0}^{\infty} \varphi^k(p).$$

We call such an algorithm $\langle p \rangle_\varphi$ a *principal* algorithm, and p its *origin*. For two algorithms (P, φ) and (Q, ψ), there exists a unique algorithm $(P, \varphi) \sqcup (Q, \psi)$ such that its state space is the disjoint union $P \sqcup Q$ and that it contains both (P, φ) and (Q, ψ) as full subalgorithms. This algorithm is called the *disjoint sum* of the algorithms (P, φ) and (Q, ψ). The disjoint sum of a family of algorithms is defined similarly. A non-empty algorithm (P, φ) is uniquely decomposed into a disjoint sum of non-empty full subalgorithms; each component of the decomposition is called a *connected component* of (P, φ). We say (P, φ) is *connected*, if (P, φ) is a connected component of itself. If (P, φ) and (P, φ') are algorithms with common state space P satisfying $\varphi'(p) \subset \varphi(p)$ for any $p \in P$, then we say that (P, φ') is a *restriction* of (P, φ). We say two algorithms (P, φ) and (Q, ψ) are *isomorphic*, if there exists a bijective map $f\colon P \longrightarrow Q$ such that $f(\varphi(p)) = \psi(f(p))$ for any $p \in P$. In that case, we write $(P, \varphi) \cong (Q, \psi)$ and call f an *isomorphism* of algorithms. The *sum*[4)] $(P, \varphi) + (Q, \psi) = (P + Q, \varphi + \psi)$ of two algorithms (P, φ) and (Q, ψ) is an algorithm whose state space $P + Q$ and branching map $\varphi + \psi$ are defined by

$$P + Q = \{(p, q) \mid p \in P, \ q \in Q\}$$

and

$$(\varphi+\psi)(p,q)=\{(p',q)\mid p'\in\varphi(p)\}\cup\{(p,q')\mid q'\in|\ q'\in\psi(q)\},\quad (p,q)\in P+Q.$$

Namely, a follower of (p,q) in $(P+Q,\varphi+\psi)$ is obtained by replacing either p or q, but not both p and q, with one of its followers. If (P,φ) and (Q,ψ) are connected, then $(P+Q,\varphi+\psi)$ is also connected. The sum of a family of algorithms is defined similarly.

2. PLAIN ALGORITHMS, DIAGRAMS, HOOKS

Since the notion of an algorithm given in the previous section is too broad for a serious study, we need to select some special classes. In this section, we introduce the notion of a plain algorithm in an axiomatic way, and state some of its basic properties. We also introduce important notions such as diagrams and hooks for a plain algorithm. See Section 8 for explicit examples related to this section.

We define two algorithms (C_0,ω_0) and (C_1,ω_1) by $C_0=\{0\}$, $\omega_0(0)=\emptyset$, and $C_1=\{0,1\}$, $\omega_1(0)=\{1\}$, $\omega_1(1)=\emptyset$. For $n\geq 2$, we define an algorithm (C_n,ω_n) as the sum

$$(C_n,\omega_n)=(C_1,\omega_1)+(C_1,\omega_1)+\cdots+(C_1,\omega_1)$$

of n copies of (C_1,ω_1). An algorithm isomorphic to (C_n,ω_n) $(n\geq 0)$ is called an *n-dimensional (hyper) cube*, or an *n-cube*, in short. A 2-cube is also called a *square*. In an n-cube (C,ω), there exists a unique point o (resp. e) such that $\omega^{-1}(o)=\emptyset$ (resp. $\omega(e)=\emptyset$); the point o (resp. e) is called the *origin* (resp. end) of C. The subset $\omega(o)$ of C is called the *basis* of C. Let (P,φ) be an algorithm, and $\beta\subset P$. If we have $b\notin\varphi^{\pm 1}(a)$ for any a, $b\in\beta$, then we say that β is *independent*. An algorithm (P,φ) satisfying the four axioms (P1)-(P4) below is called a *plain algorithm*.

(P1) Let $p\in P$, and $\beta\subset\varphi(p)$. If β is independent and consists of n elements, then there exists a unique subalgorithm of (P,φ) which is an n-cube with origin p and basis β.

(P2) Let $p,q,s\in P$. If $q\in\varphi(p)$, $s\in\varphi(q)$, and $s\notin\varphi(p)$, then there exists a unique $r\in P$ such that $\{p,q,r,s\}$ is a square.

(P3) Let $p,s_1,s_2\in P$, and $q,r_1,r_2\in\varphi(p)$. If both $\{p,q,r_1,s_1\}$ and $\{p,q,r_2,s_2\}$ are squares, then we have: $r_2\in\varphi(r_1)\Longleftrightarrow s_2\in\varphi(s_1)$.

(P4) Let $p,s\in P$, $q,r,t\in\varphi(p)$, and $t\neq q,r$. If $\{p,q,r,s\}$ is a square, then we have: $s\in\varphi^{\pm 1}(t)\Longleftrightarrow q,r\in\varphi^{\pm 1}(t)$.

Note that the axioms (P1)-(P4) are local conditions in the sense that they are only concerned with a "neighbor" of a point $p\in P$. The purpose of this paper is to survey global results (such as Theorems 4.1, 4.2, 5.1, 5.2, 7.1 below) derived from them[5)]. The axiom (P1) says that a plain algorithm is obtained by gluing n-cubes together for various n (see Fig. 1); (P2) says that a point $s\in P$ which cannot be reached from a point $p\in P$ via a single arrow, but can be reached via two consecutive arrows is an end of a square with origin p (see Fig. 2); (P3) gives a relation between two squares which have the origin and an arrow in common (see Fig. 3); (P4) gives a relation between a square and an arrow which have the origin in common (see Fig. 4; a non-oriented segment in Fig. 4 represents an arrow whose orientation is unspecified).

By (P1)-(P4), we see that the branching pattern in a plain algorithm is rather simple. Clearly, an n-cube is plain. The sum of two plain algorithms is plain. We state some of the basic properties of a plain algorithm.

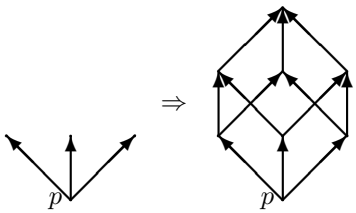

FIGURE 1. (P1) $(n = 3)$

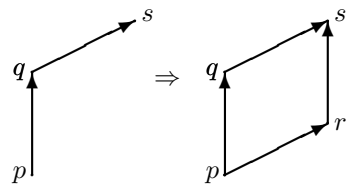

FIGURE 2. (P2)

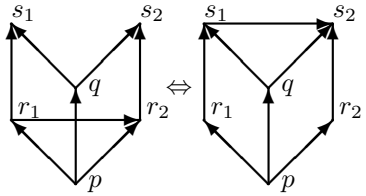

FIGURE 3. (P3)

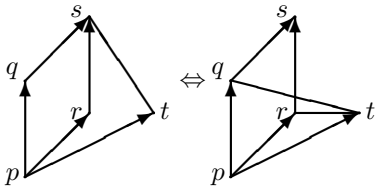

FIGURE 4. (P4)

Proposition 2.1. *Let (P, φ) be a plain algorithm. We have $p \notin \varphi^k(p)$ for any $p \in P$ and any $k \geq 1$.*

Proposition 2.2. *An end of a connected plain algorithm is unique. In particular, for any plain algorithm (P, φ) and any $p \in P$, an end of $\langle p \rangle_\varphi$ is unique.*

Proposition 2.3. *Let (P, φ) be a finitely branching plain algorithm. Let k be a positive integer. For any $p \in P$, we have:*

$$\varphi^k(p) = \emptyset \Longleftrightarrow k \geq |\varphi(p)| + 1,$$

where, for a finite set S, $|S|$ denotes the number of its elements. In particular, a finitely branching plain algorithm is finite[6]*.*

A full subalgorithm of a plain algorithm is clearly plain. In particular, a principal subalgorithm of a plain algorithm is plain. A plain algorithm may also contain a lot of plain non-full subalgorithms. For example, we have the following result.

Theorem 2.4. *Let (P, φ) be a plain algorithm. For any $p \in P$, the subalgorithms $\varphi^{\pm 1}(p)$ and $\varphi(p)$ are both plain. If e is an end of (P, φ), then the subalgorithm $\varphi^{-1}(e)$ is plain.*

The next theorem may be called the "fundamental theorem of plain algorithms".

Theorem 2.5. *Let (P, φ) be a plain algorithm. For $p \in P$, we put $Y_p = \varphi(p)$. The isomorphism class of the principal algorithm $\langle p \rangle_\varphi$ is uniquely determined by the isomorphism class of the subalgorithm Y_p of P.*

The subalgorithm Y_p mentioned in Theorem 2.5 is plain by Theorem 2.4, and is called the *diagram* of the principal plain algorithm $\langle p \rangle_\varphi$. In general, an algorithm isomorphic to the diagram of a principal plain algorithm is called a *plain diagram* (or, just, *diagram*). We define a map $H_p = H_{p,\varphi} \colon Y_p \longrightarrow 2^{Y_p}$ by

$$H_p(x) = \{x\} \cup \varphi_{Y_p}^{-1}(x) = \{x\} \cup (\varphi(p) \cap \varphi^{-1}(x)), \quad x \in Y_p. \tag{2.1}$$

We call H_p the *hook map* at p, and $H_p(x)$ the *hook* of $x \in Y_p$ at p. If $|H_p(x)|$ is finite, it is called the *length* of the hook $H_p(x)$. As explained later in Section 8, a Young diagram and its hook (see Fig. 5) can be considered as examples of a plain diagram and its hook. The set of points q such that $p \to q$ is the underlying set of the algorithm Y_p. The set Y_p equipped with the hook map H_p is equivalent to the notion of the plain diagram (Y_p, φ_{Y_p}). Hence, Theorem 2.5 says that any information about the principal plain algorithm $\langle p \rangle_\varphi$ can be read off, in principle, from the "much smaller" set Y_p and its hook map H_p[7]. Some of the main results (Theorems 4.1, 4.2, 5.1, 5.2, 7.1) of this paper may be considered as manifestations of this fundamental principle. We shall come back to this point of view a few times later.

Under the above notation, let $p \to q$. It can be shown that the diagram Y_q of $\langle q \rangle_\varphi$ is obtained by "subtracting" the hook $H_p(q)$ from the diagram Y_p of $\langle p \rangle_\varphi$. For example, an arrow $p \to q$ is irreducible if and only if $H_p(q) = \{q\}$, and, in that case, we have a natural isomorphism $Y_q \cong Y_p \setminus \{p\}$ as algorithms. Hence, the irreducible paths with origin p are in 1-1 correspondence with the sequences of successive subtractions of hooks of length 1 starting from the diagram Y_p. We omit the details for the case $|H_p(q)| > 1$.

3. Dynkin diagrams, (G, K)-quotients

In Sections 3-7, we shall describe various properties of a plain algorithm. Readers who want to see explicit examples are referred to Sections 8-11. Let (P, φ) be a connected finitely branching plain algorithm[8]. By Propositions 2.2 and 2.3, there exists a unique end e of P. By Theorem 2.4, the algorithm $\varphi^{-1}(e)$ is plain. Hence, each connected component P_{i_1} ($i_1 \in I$) of it has a unique end e_{i_1}. By Theorem 2.4, the algorithm $P_{i_1} \cap \varphi^{-1}(e_{i_1})$ is plain. Hence, each connected component $P_{i_1 i_2}$ ($i_2 \in I_{i_1}$) of it has a unique end $e_{i_1 i_2}$. Repeating the similar procedure as far as possible (may be an infinite number of times), we get connected plain subalgorithms

$$P_{i_1 i_2 \cdots i_k}, \quad i_1 \in I,\ i_2 \in I_{i_1},\ \ldots,\ i_k \in I_{i_1 i_2 \cdots i_{k-1}}, \quad k \geq 1$$

and their ends $e_{i_1 i_2 \cdots i_k}$. Clearly, $\alpha_{i_1 i_2 \cdots i_k} = (e_{i_1 i_2 \cdots i_k} \to e_{i_1 i_2 \cdots i_{k-1}})$ ($e_{i_1 i_2 \cdots i_{k-1}} = e$ if $k = 1$) is an irreducible arrow. Consider a graph[9] $D(P) = D(P, \varphi)$ with vertices $\alpha_{i_1 i_2 \cdots i_k}$ ($k \geq 1$) and edges $\{\alpha_{i_1 i_2 \cdots i_k}, \alpha_{i_1 i_2 \cdots i_{k-1}}\}$ ($k \geq 2$). Adjoining a new vertex α_* and new edges $\{\alpha_{i_1}, \alpha_*\}$ ($i_1 \in I$) to $D(P)$, we get a graph $D(P)_* = D(P, \varphi)_*$. In order to be able to read off the direction of arrows from $D(P)_*$, we denote the vertex α_* (resp. vertices other than α_*) by a white node (resp. black nodes). We call $D(P)_*$ the *Dynkin diagram* of (P, φ)[10]. See Fig. 8 for an explicit example. The Dynkin diagram does not determine the isomorphism class of the corresponding algorithm; it gives merely a coarser classification. Let P be a non-connected plain algorithm. The Dynkin diagram $D(P)_*$ of P is, by definition, the disjoint union of the Dynkin diagrams of the connected components of P. The definition of $D(P)$ is similar. If one consider a graph $E(P) = E(P, \varphi)$ with vertices $e_{i_1 i_2 \cdots i_{k-1}}$ ($k \geq 1$) and edges $\alpha_{i_1 i_2 \cdot i_k}$ (discarding the directions), then it is naturally isomorphic to $D(P)_*$.

Proposition 3.1. *Let (P, φ) be a finitely branching plain algorithm. Let $p \in P$. We denote by $E(Y_p)_*$ the graph obtained from the graph $E(Y_p)$ by adjoining a new vertex $*$ and a new edges $\{e_{Y_p, j}, *\}$ ($j \in J$), where $e_{Y_p, j}$ ($j \in J$) are the ends of Y_p.*

We have a natural graph isomorphism from $D(\langle p\rangle_\varphi)_*$ onto $E(Y_p)_*$, which sends α_* to $*$.

We denote by $\Pi(P) = \Pi(P, \varphi)$ (resp. $\Pi(P)_* = \Pi(P, \varphi)_*$) the set of vertices of $D(P)$ (resp. $D(P)_*$). Hence, we have $\Pi(P) = \Pi(P)_* \setminus \{\alpha_*\}$. We denote by $\mathbb{Z}^+\Pi(P)$ the set of formal sums of elements of $\Pi(P)$, and by $\mathcal{A}(P) = \mathcal{A}(P, \varphi)$ the set of arrows of (P, φ). We have natural inclusions $i\colon \Pi(P) \hookrightarrow \mathbb{Z}^+\Pi(P)$ and $j\colon \Pi(P) \hookrightarrow \mathcal{A}(P)$.

Proposition 3.2. *Let (P, φ) be a finitely branching plain algorithm. There exists a unique map $F\colon \mathcal{A}(P) \longrightarrow \mathbb{Z}^+\Pi(P)$ satisfying the conditions* (a)(b)(c) *below. Moreover, we have $F(p \to q) \in \Pi(P)$, if $(p \to q) \in \mathcal{A}(P)$ is irreducible.*

(a) $i = F \circ j$;

(b) *If $(p \to q)$, $(q \to r)$, $(p \to r) \in \mathcal{A}$, then we have*

$$F(p \to r) = F(p \to q) + F(q \to r);$$

(c) *If $\{p, q, r, s\}$ is a square of (P, φ) with origin p and end s, then we have*

$$F(p \to q) = F(r \to s), \quad F(p \to r) = F(q \to s).$$

Proposition 3.2 is a result of Jordan-Hölder type for an irreducible decomposition of an arrow. Under the map F, an irreducible arrow of (P, φ) corresponds to an element of $\Pi(P)$, a 'simple root' in the terminology of root systems [5]. It can also be shown that a general (non-irreducible) arrow corresponds to a 'positive root'. Later, in Section 7, we use $F(p \to q)$ as a weight of an arrow $p \to q$ when we consider a sum over paths. The following proposition indicates another application of the map F.

Proposition 3.3. *Let (P, φ) be a finitely branching plain algorithm. For any commutative semigroup G and any map $h\colon \Pi(P) \longrightarrow G$, let $\tilde{h}\colon \mathbb{Z}^+\Pi(P) \longrightarrow G$ be the extension of h to a semigroup homomorphism. Let $F\colon \mathcal{A}(P) \longrightarrow \mathbb{Z}^+\Pi(P)$ be as in Proposition* 3.2. *We define $f\colon \mathcal{A}(P) \longrightarrow G$ by $f = \tilde{h} \circ F$. For any subsemigroup K of G, we put $\mathcal{A}_K(P) = \mathcal{A}_K(P, \varphi) = f^{-1}(K) \cap \mathcal{A}(P, \varphi)$, and define a map $\varphi_{(G,K)}\colon P \longrightarrow 2^P$ by*

$$\varphi_{(G,K)}(p) = \{q \in \varphi(p) \mid (p \to q) \in \mathcal{A}_K(P, \varphi)\}.$$

The restriction $(P, \varphi_{(G,K)})$ of (P, φ) is a finitely branching plain algorithm. In particular, the algorithm $\langle p\rangle_{\varphi_{(G,K)}}$ has a unique end.

Since two algorithms $(P, \varphi_{(G,K)})$ and (P, φ) have the state space P in common, we can consider two diagrams $\varphi_{(G,K)}(p)$ and $\varphi(p) = Y_p$ for a point $p \in P$. The former is called the (G, K, h)-*quotient* (or (G, K)-*quotient*, in short) of the latter. The end of $\langle p\rangle_{\varphi(G,K)}$ is called the (G, K, h)-*core* (or (G, K)-*core* in short) of p. In the special case when $G = \langle g\rangle$ is the cyclic group of order n, h is defined by

$$h(p \to q) = g, \quad (p \to q) \in \Pi(P, \varphi),$$

and K is the trivial subgroup of G, we also use the notation φ_n for $\varphi_{(G,K)}$. The map $\varphi_n\colon P \longrightarrow 2^P$ can also be defined by

$$\varphi_n(p) = \{q \in \varphi(p) \mid |p \to q| \equiv 0 \mod n\}, \quad p \in P \tag{3.1}$$

where $|p \to q|$ denotes the number of irreducible arrows contained in a irreducible decomposition of $p \to q$. The diagram $\varphi_n(p)$ (resp. the end) of $\langle p\rangle_{\varphi_n}$, which is a special cases of the (G, K)-quotient (resp. the (G, K)-core) is also called the n-*quotient* of the original diagram $\varphi(p)$ (resp. the n-core of p). This generalizes the

notion of n-quotient (resp. n-core) in the theory of Young diagrams (see Section 8). The notion of (G, K)-quotient for a non-cyclic group G will be used, in Section 4, to analyze a plain algorithm considered as a 2-person game.

4. Plain algorithms as 2-person games

See Section 8 for explicit examples related to this section. Let (P, φ) be a finitely branching plain algorithm. Let $n \geq 2$ be a natural number. In view of the notion of n-quotient introduced in the previous section, it is natural to consider the following "n-adic expansion". We define a map $E_{\varphi}^{(n)} \colon P \longrightarrow \mathbb{N} = \{0, 1, 2, \ldots\}$ by

$$E_{\varphi}^{(n)}(p) = \sum_{i=0}^{\infty} n^i \varepsilon_{\varphi,i}^{(n)}(p), \quad p \in P, \tag{4.1}$$

where $\varepsilon_{\varphi,i}^{(n)}(p) \in \{0, 1, 2, \ldots, n-1\}$ is an analogue of the coefficient of n^i in the n-adic expansion of a natural number, and is defined by[11)]

$$\varepsilon_{\varphi,0}^{(n)} \equiv |\varphi(p)| \mod n$$

and

$$\varepsilon_{\varphi,i+1}^{(n)} = \varepsilon_{\varphi_n,i}^{(n)}, \quad i = 0, 1, 2, \ldots$$

using (3.1). We call $E_{\varphi}^{(n)}$ the *n-adic expansion function* of (P, φ). Unlike the n-adic expansion of a natural number, the value of $E_{\varphi}^{(n)}$ at $p \in P$ is, in general, depends on n. For a general n, we know very little about $E_{\varphi}^{(n)}$. But, as shown in Theorems 4.1 and 4.2 below, the 2-adic expansion function $E_{\varphi}^{(2)}$ contains decisive information of (P, φ) as a 2-person game.

Theorem 4.1. *Let (P, φ) be a finitely branching plain algorithm, and $E_{\varphi}^{(2)}$ its 2-adic expansion function. Let $p \in P$. We have:*

(i) *there exists a point $q \in \varphi(p)$ such that $E_{\varphi}^{(2)}(q) = k$ for an integer k satisfying $0 \leq k < E_{\varphi}^{(2)}(p)$;*
(ii) *$E_{\varphi}^{(2)}(p) \neq E_{\varphi}^{(2)}(p')$ for $p' \in \varphi(p)$;*
(iii) *$E_{\varphi}^{(2)}(p) = 0$, if and only if there exists a winning strategy[12)] for the second player in the 2-person game (P, φ) with opening position $p \in P$; otherwise, there exists a winning strategy for the first player.*

Theorem 4.1 (i)(ii) states that $E_{\varphi}^{(2)}(p)$ is nothing but the Sprague-Grundy number[13)] of the position $p \in P$. Perhaps it is worthwhile to remark that nim addition[14)] closely related to the Sprague-Grundy theory is apparently not needed in formulating or proving Theorem 4.1 (and Theorem 4.2 below). In a sense, getting a formula for the Sprague-Grundy number, as in Theorem 4.1, is not good enough; even if we had one, at a given winning position of one of the players, the search of next good moves for that player often needs a lot of calculation involving trial and error[15)]. Fortunately, for the game (P, φ) in Theorem 4.1, we need not rely on trial and error, since there exists a more straightforward way described in Theorem 4.2. The notion of (G, K)-quotient turns out to be very useful here. Let G be Klein's four group, which is the simplest non-cyclic group. To fix notation, we put $G = \langle a, b \rangle$ with fundamental relations $a^2 = b^2 = 1$, $ab = ba$. Since the Dynkin diagram $D(P, \varphi)_*$ is a tree, we can define, for two vertices v, w of $D(P, \varphi)_*$, the

distance $d(v,w)$ as the number of edges between them. For $\alpha \in \Pi(P,\varphi)$, we define a map $h\colon \Pi(P,\varphi) \longrightarrow G$ by

$$h(\alpha) = \begin{cases} a, & \text{if } d(\alpha,\alpha_*) \text{ is odd;} \\ b, & \text{if } d(\alpha,\alpha_*) \text{ is even.} \end{cases} \tag{4.2}$$

The group G has three non-trivial proper subgroups, which are the cyclic subgroups $\langle a\rangle$, $\langle b\rangle$ and $\langle ab\rangle$. Thus, applying the general construction in the previous section, we get the finitely branching plain algorithms $(P,\varphi_{(G,\langle a\rangle)})$, $(P,\varphi_{(G,\langle b\rangle)})$, and $(P,\varphi_{(G,\langle ab\rangle)})$, and the corresponding n-adic (in particular, 2-adic) expansion functions $E^{(n)}_{\varphi_{(G,\langle a\rangle)}}$, $E^{(n)}_{\varphi_{(G,\langle b\rangle)}}$, and $E^{(n)}_{\varphi_{(G,\langle ab\rangle)}}$ [16]. The next theorem gives an efficient method for the search of good moves. (Although we have stated the above theorem before the next one, the order of proofs are different; we first prove Theorem 4.2 below, and then prove Theorem 4.1 using Theorem 4.2.)

Theorem 4.2. *Let (P,φ) be a finitely branching plain algorithm. Let $G = \langle a,b\rangle$ be Klein's four group. Let $\varphi_{(G,\langle a\rangle)}$, $\varphi_{(G,\langle b\rangle)}$ and $\varphi_{(G,\langle ab\rangle)}$ be the branching maps from P to 2^P defined above using the map $h\colon \Pi(P,\varphi) \longrightarrow G$ given by* (4.2).

(i) *For $p \in P$, we have:*

$$E^{(2)}_{\varphi_{(G,\langle ab\rangle)}}(p) = E^{(2)}_{\varphi_{(G,\langle b\rangle)}}(p) = \sum_{i=0}^{\infty} 2^i \varepsilon^{(2)}_{\varphi,i+1}(p),$$

and

$$E^{(2)}_{\varphi_{(G,\langle a\rangle)}}(p) = \varepsilon^{(2)}_{\varphi,0}(p) + \sum_{i=1}^{\infty} 2^i \varepsilon^{(2)}_{\varphi,i+1}(p).$$

(ii) *For $p \in P$ and a non-negative integer t, we put*[17]

$$A(\varphi;p;t) = \{q \in \varphi(p) \mid E^{(2)}_{\varphi}(q) = t\}.$$

Let $t = \sum_{i=0}^{\infty} 2^i t_i$ $(t_i = 0,1)$ be the 2-adic expansion of t. We have:

$$A(\varphi;p;t) = \begin{cases} A(\varphi_{(G,\langle ab\rangle)};p;(t-t_0)/2), & \text{if } \varepsilon^{(2)}_{\varphi,0}(p) = t_0; \\ A(\varphi_{(G,\langle a\rangle)};p;(t-2t_1+t_0)/2), & \text{if } \varepsilon^{(2)}_{\varphi,0}(p) \neq t_0 \text{ and } \varepsilon^{(2)}_{\varphi,1}(p) = t_1; \\ A(\varphi_{(G,\langle b\rangle)};p;(t-t_0)/2), & \text{if } \varepsilon^{(2)}_{\varphi,0}(p) \neq t_0 \text{ and } \varepsilon^{(2)}_{\varphi,1}(p) \neq t_1. \end{cases}$$

Theorem 4.2(ii) claims that, for a given $p \in P$, the problems concerning the values taken on $\varphi(p)$ of the 2-adic expansion function $E^{(2)}_{\varphi}$ of (P,φ) can be reduced to similar problems for the 2-adic expansion functions of the three restrictions $(P,\varphi_{(G,\langle a\rangle)})$, $(P,\varphi_{(G,\langle b\rangle)})$ and $(P,\varphi_{(G,\langle ab\rangle)})$ of (P,φ). This process may be iterated, and, after a finite number of iterations, the original problems are reduced to trivial ones. In particular, by Note 12), the application of this method in the case $t = 0$ leads to an efficient method for the search of good moves. An explicit example is given in Section 8.

For a general finite algorithm viewed as a 2-person game, it is easy to see, by induction, that one of the player has a winning strategy. But, if the game is getting somewhat complex, then to decide which player has a winning strategy and to describe an explicit winning strategy soon require too much computation even for a computer. The games we considered in this section are rather exceptional, because we can rapidly achieve the same purpose using Theorems 4.1 and 4.2. This fact may be seen as a manifestation of the fundamental principle mentioned after Theorem

2.5. A plain algorithm as a 2-person game might be considered as an example of *completely solvable games* in the sense that it can be analyzed completely and efficiently. There remains much to be done in this area, as can be seen, for example, from a recent work of T. Kayada [10] on the 'Sato-Welter game of height k'.

5. Probabilistic algorithms and Peterson's hook-length formula

In this section, we reformulate the main result of S. Okamura [15] in terms of the theory of plain algorithms. An explicit example is given in Section 10. Let (P, φ) be a finitely branching plain algorithm. We define a map $\varphi_{\mathrm{irred}} \colon P \longrightarrow 2^P$ by

$$\varphi_{\mathrm{irred}}(p) = \{q \in \varphi(p) \mid p \to q \text{ is an irreducible arrow}\}.$$

By Proposition 2.3, (P, φ) is finite. Hence, $(P, \varphi_{\mathrm{irred}})$ is also a finite algorithm. The ends of (P, φ) coincide with those of $(P, \varphi_{\mathrm{irred}})$. If $p \in P$ is not an end, we define a probabilistic measure on $\varphi_{\mathrm{irred}}(p)$ as follows. Adjoining a new point $*$ (see Proposition 3.1) to the set $Y_p = \varphi(p) \neq \emptyset$, we define $Y_p^* = \{*\} \sqcup \varphi(p)$. We also define $K_p \colon Y_p^* \longrightarrow 2^{Y_p^*}$ by putting, for $x \in Y_p^*$,

$$K_p(x) = \begin{cases} \varphi(p), & \text{if } x = *; \\ \varphi(p) \cap \varphi^{-1}(x) = H_p(x) \setminus \{x\}, & \text{if } x \neq *. \end{cases}$$

In the algorithm (Y_p^*, K_p), we consider a path with origin $*$

$$* = x_0 \to x_1 \to x_2 \to \cdots \to x_m \to \cdots,$$

where, for any $m \geq 0$ such that x_m is not an end of (Y_p^*, K_p), a next point x_{m+1} is selected from the set $K_p(x_m)$ uniform randomly. After a finite number of steps, the algorithm terminates. This probabilistic algorithm (in the sense of Section 1) selects an end of (Y_p^*, K_p) with some probability. Since the set of ends of (Y_p^*, K_p) coincides with $\varphi_{\mathrm{irred}}(p)$, the probabilistic algorithm (Y_p^*, K_p) defines a probabilistic measure on $\varphi_{\mathrm{irred}}(p) \neq \emptyset$.
For any $p \in P$, the probabilistic algorithm $(P, \varphi_{\mathrm{irred}})$ with respect to the above measure selects a path

$$p = p_0 \to p_1 \to p_2 \to \cdots \to p_l \to \cdots \to p_k = e \quad (e \text{ is an end of } (P, \varphi)) \tag{5.1}$$

in $(P, \varphi_{\mathrm{irred}})$ with some probability. By Proposition 2.3, the length k of a path (5.1) is $|\varphi(p)|$.

Theorem 5.1. (Okamura [15]) *Let (P, φ) be a finitely branching plain algorithm. Under the probabilistic algorithm $(P, \varphi_{\mathrm{irred}})$, the probability with which a path* (5.1) *is selected depends only on its origin p, and is equal to*

$$\frac{\prod_{x \in Y_p} |H_p(x)|}{|Y_p|!}. \tag{5.2}$$

From Theorem 5.1, we immediately get the following result.

Theorem 5.2. *Let (P, φ) be a finitely branching plain algorithm. For $p \in P$, the number of paths in $(P, \varphi_{\mathrm{irred}})$ of the form* (5.1) *is equal to*

$$\frac{|Y_p|!}{\prod_{x \in Y_p} |H_p(x)|}. \tag{5.3}$$

Theorem 5.2 is equivalent to the 'hook-length formula' due to D. Peterson for the number of reduced expressions of a miniscule element[18)] of a Weyl group. (See [1], where the hook-length formula of Peterson is stated without proof.) As far as the present author knows, the first written proof of Peterson's hook-length formula appeared in the master's thesis [15] of Okamura. In [15], Theorems 5.1 and 5.2 are stated and proved using the notion of a d-complete poset, which was introduced by G. Proctor [16] [17] as a combinatorial counterpart of reduced expressions of a miniscule element. Further developments of the results in this section are given in Section 7.

The author does not know whether Theorem 5.1 is an isolated result peculiar to the probabilistic algorithm mentioned above or just a tiny part of a bigger picture.

6. Classification of principal plain algorithms and their realization using elements of Coxeter groups

By Theorem 2.5, the classification of principal plain algorithms is reduced to that of plain diagrams. Since plain diagrams are very special plain algorithms, this is an essential reduction of the classification problem. On the other hand, among plain diagrams, principal plain diagrams (namely, plain diagrams which are principal algorithms) are of fundamental importance. Using Theorem 2.5 again, the classification of principal plain diagrams is reduced to that of *basic plain diagrams* (namely, diagrams of principal plain diagrams). Since basic plain diagrams are rather special plain diagrams, the classification is further simplified. This process can be iterated. If we start with a finitely branching principal plain algorithm, then, after a finite number of iterations of the reduction process, we eventually arrive at the empty algorithm $(\emptyset, \emptyset)$. Reversing this, and using Proposition 6.1 below, we can construct and classify the finitely branching plain algorithms.

Proposition 6.1. *Let (Y, μ) be a plain diagram, and (Z, ν) a basic plain diagram. We have:*

(i) *If $x, y_1, y_2, y_3 \in Y$ are such that $y_i \notin \mu(y_j)$, $y_i \in \mu^{\pm 1}(x)$ and $y_i \neq y_j$ $(i \neq j)$ for any $1 \leq i, j \leq 3$, then $\{x, y_1, y_2, v\}$ is not a square for any $v \in Y$. In particular, a plain diagram does not contain a 3-cube.*

(ii) *If $v, y_1, y_2 \in Y$ satisfy $v \in \mu(y_1) \cap \mu(y_2)$, then a point $x \in Y$ such that $\{x, y_1, y_2, v\}$ is a square is unique.*

(iii) *If $z_1, z_2, z_3 \in Z$ are such that $z_i \neq z_j$ $(i \neq j)$, then $\{z_1, z_2, z_3\}$ is not independent.*

For the classification and the realization of finitely branching plain algorithms, it is convenient to use the terminology of the theory of Coxeter groups [5]. Let D_* be the Dynkin diagram of a finitely branching principal plain algorithm $\langle p \rangle_\varphi$. Let V be the real vector space of the formal linear combinations of the set Π_* of vertices of D_*. We define an inner product on V, by putting

$$(\alpha, \beta) = \begin{cases} 1, & \text{if } \alpha = \beta; \\ -1/2, & \text{if there exists an edge connecting } \alpha \text{ and } \beta; \\ 0, & \text{if there exists no edge connecting } \alpha \text{ and } \beta \end{cases} \tag{6.1}$$

for $\alpha, \beta \in \Pi_*$. For $\alpha \in \Pi = \Pi_* \setminus \{\alpha_*\}$, we define $s_\alpha \in GL(V)$ by

$$s_\alpha v = v - 2(\alpha, v)\alpha, \qquad v \in V.$$

The subgroup $W = \langle S \rangle$ generated by $S = \{s_\alpha \mid \alpha \in \Pi\}$ is a Coxeter group. Let e be an end of $\langle p \rangle_\varphi$. For an irreducibly decomposed path

$$p = p_0 \to p_1 \to p_2 \to \cdots \to p_l = e \tag{6.2}$$

with origin p, we put, using the map F in Proposition 3.2,

$$\alpha_i = F(p_{i-1} \to p_i) \ (\in \Pi), \qquad 1 \leq i \leq l$$

and define an element

$$w = s_{\alpha_1} s_{\alpha_2} \cdots s_{\alpha_l} \tag{6.3}$$

of W corresponding to the path (6.2). It can be shown that (6.3) is a reduced decomposition, and that the element w depends only on $\langle p \rangle_\varphi$ and does not depend on how we select an irreducibly decomposed path (6.2). Conversely, starting from such an element $w \in W$, we can recover the algorithm $\langle p \rangle_\varphi$. Let T be the set of elements of W which are conjugate with elements of S. An element of T (resp. S) is called a *reflection* (resp. *simple reflection*) of W. We put $R = W\alpha_*$. For two elements γ and δ of R, we write $\delta < \gamma$ if

$$\gamma - \delta \in \mathbb{Z}^+\Pi, \quad \gamma \neq \delta.$$

Let J be the set of elements β of Π orthogonal to α_*, and $W_J = \langle s_\theta \mid \theta \in J \rangle$ the corresponding parabolic subgroup of W. By a standard result of the theory of Coxeter groups, each residue class $vW_J \in W/W_J$ has a unique representative v^J such that $l(v^J s_\theta) = l(v^J) + 1$ for any $\theta \in J$ (l is the 'length function' on W). We define a map $\rho\colon W\alpha_* \longrightarrow 2^{W\alpha_*}$ by

$$\rho(\gamma) = \{t\gamma \mid t \in T,\, t\gamma < \gamma\}, \quad \gamma \in W\alpha_*,$$

and define $\sigma\colon W/W_J \longrightarrow 2^{W/W_J}$ by

$$\sigma(vW_J) = \{tvW_J \mid t \in T,\, l((tv)^J) < l(v^J)\}, \quad vW_J \in W/W_J.$$

Two algorithms $(W\alpha_*, \rho)$ and $(W/W_J, \sigma)$ are naturally isomorphic. The next theorem shows how to recover the algorithm $\langle p \rangle_\varphi$ from an element $w \in W$ defined by (6.3) using a path (6.2) in $\langle p \rangle_\varphi$.

Theorem 6.2. *Under the above notation, we have*

$$\langle p \rangle_\varphi \cong \langle w\alpha_* \rangle_\rho \cong \langle wW_J \rangle_\sigma.$$

The next theorem gives a characterization of an element w of a Coxeter group obtained by (6.2) and (6.3).

Theorem 6.3. *Let D_* be an arbitrary tree graph. Let Π_* be the set of vertices of D_*. Let $\alpha_* \in \Pi_*$. We put $\Pi = \Pi_* \setminus \{\alpha_*\}$. From these data, we define V, $s_\alpha \in GL(V)$ $(\alpha \in \Pi)$, $S = \{s_\alpha \mid \alpha \in \Pi\}$, $W = \langle S \rangle$, T, and J as above. The set of elements $w \in W$ which are defined by* (6.2) *and* (6.3) *from a finitely branching principal plain algorithm $\langle p \rangle_\varphi$ whose Dynkin diagram is D_* coincide with the set of elements $w \in W$ satisfying one of the equivalent conditions* (a) *and* (b) *given below.*

(a) *If $w = s_{\alpha_1} s_{\alpha_2} \cdots s_{\alpha_l}$ is a reduced decomposition in elements of S, then we have:*

$$s_{\alpha_i} s_{\alpha_{i+1}} \cdots s_{\alpha_l} \alpha_* - s_{\alpha_{i+1}} \cdots s_{\alpha_l} \alpha_* = \alpha_i, \quad 1 \leq i \leq l.$$

(b) *For an irreducible arrow $uW_J \to vW_J$ in the algorithm $\langle wW_J \rangle_\sigma$, there exists an $s \in S$ such that $su^J = v^J$.*

The condition (a) in Theorem 6.3 is essentially the same as the definition [1] of a miniscule element (restricted to the case of a Weyl group corresponding to a simply-laced Dynkin diagram) due to Peterson. Hence, a finitely branching principal plain algorithm can be realized by Theorem 6.2 from a miniscule element w of a Weyl group corresponding to a simply-laced Dynkin diagram; moreover, the classification of finitely branching principal plain algorithms is reduced to that of w. Since such elements w are already classified by Proctor [16], we can say that finitely branching principal plain algorithms are also classified. We can also use an element w of a Coxeter group W corresponding to a non-simply-laced Coxeter graph satisfying the condition (b)[19] in Theorem 6.3 to construct a finitely branching principal plain algorithm. If W is crystallographic, such elements w essentially coincide with miniscule elements of W classified by J. R. Stembridge [22]. We omit the details of the non-crystallographic case.

7. Colored hook formula

In this section, we reformulate the main result of K. Nakada [12] in terms of the theory of plain algorithms. An explicit example is given in Section 11. We observe that Theorems 5.1 and 5.2 state properties of a plain diagram Y_p. As mentioned in Section 6, the study of plain diagrams is reduced to that of basic plain diagrams. From this point of view, Okamura and the author examined the paper [15] and its origin [3] (see Section 10) and found that a rational function identity that plays a fundamental role in those papers reflects important properties of basic plain diagrams. Since such an identity can be formulated in terms of plain diagrams, Okamura and the author conjectured that it must be true for a plain diagram (and not just for a basic plain diagram). Soon after that, Nakada, using his own formulation, proved the conjecture, which is Theorem 7.1 below.

Let (P, φ) be a finitely branching plain algorithm. Let $p \in P$. For a path

$$\mathbf{p}\colon p = p_0 \to p_1 \to p_2 \to \cdots \to p_l \quad (l \geq 0)$$

with origin p, we define, using the map $F\colon \mathcal{A} \longrightarrow \mathbb{Z}^+\Pi(\langle p \rangle_\varphi)$ in Proposition 3.2, a polynomial

$$\mathcal{F}(\mathbf{p}) = \prod_{k=1}^{l} \{F(p_0 \to p_1) + F(p_1 \to p_2) + \cdots + F(p_{k-1} \to p_k)\}$$

in the elements of $\Pi(\langle p \rangle_\varphi)$. (In the case $l = 0$, we understand that $\mathcal{F}(\mathbf{p}) = 1$.) In Nakada [12], the notion of 'predominant integral weight' is introduced for a non-simply-laced Dynkin diagram, and plays an important role in formulating and proving the main result. In terms of the theory of plain algorithms, the result of Nakada [12] (in the simply-laced case) can be stated as follows.

Theorem 7.1. (Nakada [12]) *Let (P, φ) be a finitely branching plain algorithm. For $p \in P$, under the above notation, we have the following 'colored hook formula':*

$$\sum_{\mathbf{p}} \frac{1}{\mathcal{F}(\mathbf{p})} = \prod_{q \in Y_p} \left(1 + \frac{1}{F(p \to q)}\right), \tag{7.1}$$

where the sum on the left hand side is taken over the set of paths $\mathbf{p}$ in (P, φ) with origin p.

The left hand side of the equality (7.1) is a sum over a "big" set related to the algorithm $\langle p \rangle_\varphi$, whereas the right hand side is a product over a "small" set related to the diagram Y_p. Thus, Theorem 7.1 may be seen as another manifestation of the fundamental principle mentioned after Theorem 2.5.
Taking the sum of the terms of degree $-|Y_p|$ of both hand side of (7.1) and specializing every element of $\Pi(\langle p \rangle_\varphi)$ to 1, we get Theorem 5.2 again. When (P, φ) is a plain diagram, a result equivalent to Theorem 7.1 was obtained in Okamura [15] using case-by-case checks partly carried out by a computer.

8. The Nakayama algorithm

So far, in the first half (Sections 1-7) of the paper, we described a general theory of plain algorithms. From now on, in the second half (Sections 8-11), we shall apply our theory to a typical example, i.e. the Nakayama algorithm. The present section corresponds to the general theory in Sections 2-3.

We briefly review the historical background. By the works of G. Frobenius and A. Young around 1900, it has been known that the equivalence classes of the irreducible representations (over a field of characteristic 0) of the symmetric group S_n are in 1-1 correspondence with the Young diagrams with n boxes. (See, for example, [6].) In 1940-41, T. Nakayama [14] studied[20] the reduction modulo a prime p of the irreducible representations of S_n. Since the reduction modulo p of an object is, more or less, to look at the remainder of the object when it is divided by p, it seems necessary, for the study of the reduction modulo p of an irreducible representation of S_n, to consider the remainder of "$Y \div p$" for a Young diagram Y. Although this is merely a thought of the present author after reading Nakayama's paper, Nakayama [14], in fact, defined a notion analogous to the remainder for a Young diagram, and introduced the notion of hooks of a Young diagram precisely for that purpose. For a Young diagram Y, the *hook* $H(x) = H_Y(x)$ of a box $x \in Y$ is the set of boxes of Y directly below or to the right of x, including x itself. In the case when Y and x are as in Fig. 5, $H_Y(x)$ consists of the boxes y_1, y_2, ..., y_5 and x itself.

$Y =$

a_1								
a_2								
a_3		x	y_1	y_2	y_3			
a_4		y_4						
a_5		y_5						
a_6								

FIGURE 5

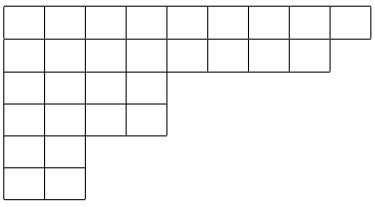

FIGURE 6. $Y - H(x)$

Let H be a hook of a Young diagram Y. If we remove H from Y and push the boxes that lie below or to the right of H upward and leftward, then we get a new Young diagram Y'. We write $Y' = Y - H$ and say that Y' is obtained by *subtracting* H from Y. Compare Fig. 5 and Fig. 6. The number of boxes in a hook H is called the *length* of the hook H and is denoted by $|H|$. The length $|H(x)|$ of the hook $H(x)$ in Fig. 5 is 6. In general, if H is a hook of a Young diagram Y of length kn with positive integers k and n, then we have:

$$Y - H = (\cdots((Y - H_1) - H_2)\cdots) - H_k,$$

where, for $1 \leq i \leq k$, H_i is a suitable hook of $(\cdots((Y - H_1) - H_2)\cdots) - H_{i-1}$ of length n. Starting from a given Young diagram, and subtracting hooks whose lengths are multiple of n successively, we eventually get a Young diagram $c_n(Y)$ that has no hook of length n. As shown in [14], such a Young diagram $c_n(Y)$, which may be the empty Young diagram $\emptyset$, is independent of the subtraction process, and depends only on Y and n. We call $c_n(Y)$ the *n-core* of Y, which may be considered as the remainder of "$Y \div n$"[21].

Nakayama [14] gave another method that is, in some cases, more convenient in describing the hook subtraction. The Young diagram Y in Fig. 5 is often identified with the non-increasing positive integer sequence (partition) $(9, 8, 6, 5, 5, 2)$, which indicates that the first row of Y consists of 9 boxes, the second row 8, the third row 6, ... The same diagram is also identified with the set of positive integers $\{14, 12, 9, 7, 6, 2\} = \{|H(a_i)| \mid 1 \leq i \leq 6\}$, where, for $1 \leq i \leq 6$, the box in the first column and the i-th row of Y is denoted by a_i; this latter numbers are called, in [14], the *β-numbers* of Y. The β-numbers of the Young diagram $Y - H(x)$ in Fig. 6 are $\{14, 12, 7, 6, 3, 2\}$, which can be obtained from those of Y by deleting 9 and adjoining 3. Moreover, the difference $9 - 3 = 6$ is equal to the length of the hook $H(x)$. In general, if H is a hook of a Young diagram Y, then the β-numbers of $Y - H$ are obtained from those of Y by deleting a number s from the β-numbers of Y and adjoining a number s' smaller than s and not contained in the β-numbers of Y; moreover, we always have $s - s' = |H|$. The converse of this is also true. This process can be visualized, if we use a board shown in Fig. 7. This board consists of squares that are in 1-1 correspondence with $\mathbb{N} = \{0, 1, 2, \ldots\}$. A stone is placed at each square corresponding to a member of the β-numbers of a given Young diagram Y. Subtracting a hook from Y exactly corresponds to moving a stone on the board to an unoccupied square with a smaller numbering. In Nakayama [14], although such a board does not appear, an equivalent statement in terms of the β-numbers mentioned above is given.

We fix a positive integer l. Let P be the set of possible arrangements of l stones,

p =	0	1	②	3	4	5	⑥	⑦	8	⑨	10	11	⑫	13	⑭	15	16	17

FIGURE 7. a board and stones

at most one for each square, on the board as in Fig. 7. Let $\varphi\colon P \longrightarrow 2^P$ be defined by

$$\varphi(p) = \{(p \setminus \{s\}) \cup \{s'\} \mid s \in p,\ s' \in \mathbb{N} \setminus p,\ 0 \leq s' < s\}, \quad p \in P,$$

where $p \in P$ is identified with the corresponding subset of $\mathbb{N}$. Let Q be the set of Young diagrams whose rows are at most l. Let $\psi\colon Q \longrightarrow 2^Q$ be defined by

$$\psi(Y) = \{Y - H(x) \mid x \in Y\}, \quad Y \in Q.$$

Two algorithms (P, φ) and (Q, ψ) are isomorphic under an isomorphism indicated above[22]. We call the isomorphism class of these algorithms *the Nakayama algorithm*, and (P, φ) and (Q, ψ) *the one-line realization*[23] and *the Young realization* of the Nakayama algorithm, respectively.

It is easy to see that the one-line realization (P, φ) of the Nakayama algorithm is a finitely branching plain algorithm[24]. Hence, by the general theory in Section 2, we have, for $p \in P$, the notion of the diagram of $\langle p \rangle_\varphi$, and that of hooks at

p. In the present case, the diagram of $\langle p\rangle_\varphi$ coincides with the Young diagram[25] Y corresponding to p, and hooks at p coincide with hooks of Y. (For example, if $p \in P$ is the state corresponding to Fig. 7, then the diagram of $\langle p\rangle_\varphi$ coincide with the Young diagram Y in Fig. 5.) More precisely, if we consider a Young diagram Y as the set of boxes of it, and define a map $\mu\colon Y \longrightarrow 2^Y$ by

$$\mu(x) = \{y \in Y \mid x \in H_Y(y),\ y \neq x\}, \quad x \in Y, \tag{8.1}$$

then the diagram of $\langle p\rangle_\varphi$ is isomorphic to (Y, μ). Moreover, if we take a $q \in P$ such that $p \to q$, then the diagram of $\langle q\rangle_\varphi$ is isomorphic to a Young diagram obtained by subtracting a hook from Y. Thus, the correspondence between the one-line realization and the Young realization of the Nakayama algorithm (in particular, the notion of a Young diagram and its hooks, and also that of the subtraction of a hook) is contained in the general theory of plain algorithms as a special case.

Let $p \in P$ be the state depicted in Fig. 7. The end of the algorithm $\langle p\rangle_\varphi$ is $e = \{0, 1, 2, 3, 4, 5\}$[26]. We see that the algorithm $\langle p\rangle_\varphi \cap \varphi^{-1}(e)$ is connected with end $f = \{0, 1, 2, 3, 4, 6\}$. Moreover, $\langle p\rangle_\varphi \cap \varphi^{-1}(e) \cap \varphi^{-1}(f)$ has two connected components whose ends are $g = \{0, 1, 2, 3, 4, 7\}$ and $h = \{0, 1, 2, 3, 5, 6\}$, ... Hence, if we put $\alpha_1 = (e \leftarrow f)$, $\alpha_2 = (f \leftarrow g)$, $\beta_1 = (f \leftarrow h)$, ..., then the Dynkin diagram $D(\langle p\rangle_\varphi)_*$ as defined in Section 3 is given by Fig. 8. We also defined, in Section 3, a map F that sends an arrow of $\langle p\rangle_\varphi$ to an element of $\mathbb{Z}^+\Pi(\langle p\rangle_\varphi) = \sum_{i=1}^{9} \mathbb{Z}^+\alpha_i + \sum_{j=1}^{5} \mathbb{Z}^+\beta_j$. In the Young realization, an arrow whose origin is a Young diagram Y corresponds to a subtraction process of a hook H from Y. We associate with each box of Y an element of $\Pi(\langle p\rangle_\varphi)$, i.e. a vertex of $D(\langle p\rangle_\varphi)_*$ other than α_*. If Y is as in Fig. 5, the correspondence is depicted in Fig. 9. It should be clear from Fig. 8 and Fig. 9 how the correspondence is given. Under this notation, $F(Y \to (Y - H))$ is equal to the sum of elements of $\Pi(\langle p\rangle_\varphi)$ corresponding to the boxes in H. In particular, if H_{nw} is the hook of the box at the north-west corner of Y in Fig. 9, then we have

$$F(Y \to (Y - H_{\mathrm{nw}})) = \sum_{i=1}^{9} \alpha_i + \sum_{j=1}^{5} \beta_j.$$

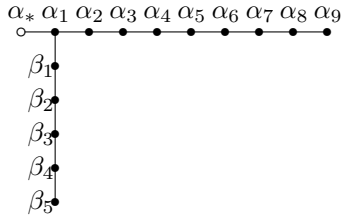

FIGURE 8. $D(\langle p\rangle_\varphi)_*$

α_1	α_2	α_3	α_4	α_5	α_6	α_7	α_8	α_9
β_1	α_1	α_2	α_3	α_4	α_5	α_6	α_7	
β_2	β_1	α_1	α_2	α_3	α_4			
β_3	β_2	β_1	α_1	α_2				
β_4	β_3	β_2	β_1	α_1				
β_5	β_4							

FIGURE 9

For a positive integer n, we define $\varphi_n\colon P \longrightarrow 2^P$ by

$$\varphi_n(p) = \{(p \setminus \{s\}) \cup \{s'\} \mid s \in p,\ s' \in \mathbb{N} \setminus \{p\},\ 0 \leq s' < s,\ s - s' \equiv 0 \mod n\}$$

for $p \in P$. The restriction (P, φ_n) of (P, φ) is plain. If we identify, for $p \in P$, the diagram of $\langle p\rangle_\varphi$ with the corresponding Young diagram Y, then the diagram of $\langle p\rangle_{\varphi_n}$ coincides with the subalgorithm

$$\varphi_n(p) = \{x \in Y \mid |H_Y(x)| \equiv 0 \mod n\},$$

which is the n-quotient of Y as defined in Section 3, and is isomorphic to a disjoint sum of n Young diagrams.
The notion of n-quotient for a Young diagram was introduced by G. de B. Robinson and used in proving Nakayama's conjecture[27] in modular representation theory of the symmetric groups. See [6] for the definitions and applications of the notions of cores and quotients for a Young diagram.

9. The Sato-Welter game

Here we illustrate the general theory in Section 4 in the case of the Nakayama algorithm. The Nakayama algorithm considered as a 2-person game is known[28] by the name of 'Welter's game', since C.P. Welter [23] first gave the fundamental result (the algebraic formula for the Sprague-Grundy number at a given position) of this game. J.H. Conway [2, Ch. 13] gave an alternative proof of Welter's result. Welter [23] and Conway [2] describe the game only in the one-line realization and do not mention about the Young realization. M. Sato [21], independently, got the same result as Welter [23], but it was published much later in Japanese. According to Sato [20], Sato's investigation was influenced by Nakayama [14]. In fact, Sato [18] [19] described the game in two ways, the one-line realization and the Young realization, and remarked that the formula giving the Sprague-Grundy number of a position of this game can be written in several seemingly different ways including the formula (9.1) below and the one given by Welter. He also pointed out that, in the Young realization, the formula can be written in a way similar to the hook-length formula (see Section 10) for the degree of an irreducible representation of a symmetric group[29]. The author of the present paper believes that the Young realization is essential for the full understanding of this game[30], and, hence, would like to call it (i.e., the Nakayama algorithm as a 2-person game) the *Sato-Welter game*.

The purpose of this section is to give a complete analysis of the Sato-Welter game from a point of view rather different from that of Welter, Sato and Conway. Our main result in this direction (Theorems 4.1 and 4.2) gives not only a formula for the Sprague-Grundy number, but also an efficient algorithm for detecting the next good moves at each position for a class of games that includes the Sato-Welter game as a special case.
Let $Y_{(n)}$ be the n-quotient of the Young diagram Y. If we put

$$\varepsilon_i^{(2)}(Y) \equiv |Y_{(2^i)}| \mod 2, \quad \varepsilon_i^{(2)}(Y) = 0,\ 1,$$

the value of the 2-adic expansion function $E^{(2)}$ at the Young diagram Y is given by

$$E^{(2)}(Y) = \sum_{i=0}^{\infty} 2^i \varepsilon_i^{(2)}(Y). \tag{9.1}$$

For example, for the Young diagram Y in Fig. 5, we have $|Y| = 35$, $|Y_{(2)}| = 17$, $|Y_{(2^2)}| = 7$, $|Y_{2^3)}| = 2$, $|Y_{(2^i)}| = 0$ ($i \geqq 4$), and, hence, $E^{(2)}(Y) = 1 + 2 + 2^2 = 7$. Hence, by Theorem 4.1, the first player has a winning strategy for the Sato-Welter game with opening position Y[31].

Let us explain how the first player can use Theorem 4.2 to find the next good moves at the position Y in Fig. 5. Let $D(\langle p \rangle_\varphi)_*$ be the Dynkin diagram in Fig. 8. Under the map h defined by (4.2), each element of $\Pi(\langle p \rangle_\varphi)$, i.e. each vertex of the Dynkin diagram $D(\langle p \rangle_\varphi)_*$ other than α_*, corresponds to an element, a or b, of

Klein's four group $G = \langle a, b\rangle$ ($a^2 = b^2 = 1$, $ab = ba$). Composing $h\colon \Pi(\langle p\rangle_\varphi) \longrightarrow G$ with the map from Y to $\Pi(\langle p\rangle_\varphi)$ depicted in Fig. 8 and Fig. 9, we get a map g from Y to $\{a, b\}$. In Fig. 10, the value of g at each box of Y is given. (The value of g for the box at the north-west corner of Y is a, and $g(x) \neq g(y)$ for each pair (x, y) of distinct boxes x and y with a common edge.) We also define $d\colon Y \longrightarrow G$ by

$$d(x) = \prod_{y \in H(x)} g(y).$$

a	b	a	b	a	b	a	b	a
b	a	b	a	b	a	b	a	
a	b	a	b	a	b			
b	a	b	a	b				
a	b	a	b	a				
b	a							

FIGURE 10

ab	a	b	ab	a	ab	1	b	a
1	b	a	1	b	1	ab	a	
b	1	ab	b	1	b			
a	ab	1	a	ab				
ab	a	b	ab	a				
ab	a							

FIGURE 11

In Fig. 11, the value of d at each box of Y is given. By Note 12), a good move for the first player is to subtract from Y a hook H such that $E^{(2)}(Y - H) = 0$. Applying the third case of Theorem 4.2 (ii) with $E^{(2)}(p)(= E^{(2)}(Y)) = 7$ and $t = 0$, we are led to look at the $(G, \langle b\rangle)$-quotient of Y:

$$\varphi_{(G,\langle b\rangle)}(p) = \{\, x \in Y \mid d(x) = 1 \ \text{ or } \ b \,\}. \tag{9.2}$$

		a				b	a	
a	b		a	b	a			
b	a		b	a	b			
		b						
		a						

FIGURE 12

		a				ab	a	
ab	a		1	b	ab			
b	1		a	ab	b			
		ab						
		a						

FIGURE 13

If we denote the right hand side of (9.2) by Y', then Y' is equal to the set of boxes in Fig. 11 containing 1 or b, and, as an algorithm, is isomorphic to the disjoint sum of the 2×5 rectangular Young diagram and a hook-shaped Young diagram of length 5. We repeat for Y' the same calculation done above for Y. The hook $H_{Y'}(y)$ of $y \in Y'$ in Y' is defined by the general formula (2.1) (replacing φ with $\varphi_{(G,\langle b\rangle)}$) and is equal to $H_Y(y) \cap Y'$. Applying the third case of Theorem 4.2 (ii) with $E^{(2)}(p)(= E^{(2)}(Y')) = 3$ and $t = 0$, we are led to look at the $(G, \langle b\rangle)$-quotient of Y'. In Fig. 12, we attach to each box of Y' an element, a or b, of G, as was done for Y in Fig. 10[32]. In Fig. 13, we attach to each box of Y' the product of elements contained in the hook of that box, as was done in Fig. 11 for Y. The $(G, \langle b\rangle)$-quotient Y'' of Y' is equal to the set of boxes containing 1 or b in Fig. 13, and is isomorphic to the disjoint sum of the 1×3 and 1×2 rectangular Young diagrams. We can again apply Theorem 4.2 (ii) to Y''. But, since, as a position of a 2-person game, Y'' is isomorphic to the Nim position with heaps of size 3 and 2, it is evident from the well-known theory of Nim (see, e.g. [4, 9.8]) that a unique

good move for the first player (at the opening position Y'') is to subtract from Y'' the hook of the box in the third row and the sixth column in Fig. 13. Hence, if Y is the opening position, a unique good move for the first player is to subtract from Y the hook of the same box.

10. The Greene-Nijenhuis-Wilf algorithm

This section corresponds to the general theory in Section 5. For a given Young diagram Y, we consider the following probabilistic algorithm:
We first select, uniform randomly, a box x of a Young diagram Y. Next we select, uniform randomly, a box y of $H_Y(x) \setminus \{x\}$, if $\{x\} \subsetneqq H_Y(x)$. If $\{y\} \subsetneqq H_Y(y)$, we further select, uniform randomly, a box of $H_Y(y) \setminus \{y\}$. Repeating this, we eventually arrive at a box z such that $H_Y(z) = \{z\}$, i.e. a box z whose hook is of length 1. Let $X = Y \setminus \{z\}$, which is also a Young diagram. We repeat the whole process taking X instead of Y. The algorithm terminates when we arrive at the empty diagram $\emptyset$.
This probabilistic algorithm, which is due to C. Greene, A. Nijenhuis and H. S. Wilf [3], selects a vanishing process of a Young diagram Y, which is equivalent to a standard Young tableau [6] of shape Y.

Theorem 10.1. (Greene, Nijenhuis, and Wilf [3]) *The Greene-Nijenhuis-Wilf algorithm generates each standard Young tableau of shape Y with uniform probability $\Pi_{x\in Y}|H(x)|/n!$. In particular, the number of standard tableaux with shape Y is equal to the inverse $n!/\Pi_{x\in Y}|H(x)|$ of the above probability.*

The latter half of Theorem 10.1 gives a proof for the famous hook-length formula (see, e.g. [6, 2.3] [11, 5.1.4]) for the number of the standard Young tableaux of a given shape, or, equivalently, for the degree of the corresponding irreducible representation of a symmetric group. Theorem 5.1 in Section 5 is a natural generalization of Theorem 10.1.

11. Colored hook formula for the Nakayama algorithm

This section corresponds to the general theory in Section 7. Let (Q, ψ) be the Young realization (see Section 8) of the Nakayama algorithm. A path

$$\mathbf{p}\colon Y = Y_0 \to Y_1 \to Y_2 \to \cdots \to Y_l, \quad 0 \le l \le |Y| \tag{11.1}$$

in (Q, ψ) is a sequence of Young diagrams Y_i $(0 \le i \le l)$ such that, for $1 \le i \le l$, $Y_i = Y_{i-1} - H_{i-1}$ for a hook H_{i-1} of Y_{i-1}. As in Fig. 9, we attach a variable to each box of the hook H_{nw} of the box at the north-west corner of Y; if a box y of Y lies to the south-east direction of a box x in H_{nw}, then we attach to y the same variable as attached to x. For a hook H of Y or of a subdiagram of Y, we denote by $[H]$ the sum of variables attached to the boxes of H. To a path $\mathbf{p}$ of (Q, ψ) given by (11.1), we attach the polynomial

$$[\mathbf{p}] = \prod_{k=0}^{l-1} (\sum_{j=0}^{k} [H_j]).$$

(If $\mathbf{p}$ is of length 0, we put $[\mathbf{p}] = 1$.) The colored hook formula for (Q, ψ) is given as follows:

$$\sum_{\mathbf{p}} \frac{1}{[\mathbf{p}]} = \prod_{x\in Y} \left(1 + \frac{1}{[H_Y(x)]}\right), \tag{11.2}$$

where the sum on the left hand side is taken over the set of paths $\mathbf{p}$ in (Q, ψ) whose origin is a given Young diagram Y. In Nakada [12], the formula (11.2) is written explicitly for the case when Y is the 2×2 Young diagram; even in that case, the left hand side of (11.2) is a rather long sum.

Let X be an $m \times n$ rectangular Young diagram, which we consider as an algorithm (see (8.1)). Let p be the box at the south-east corner of X; X is a principal plain algorithm generated by p. Let Y_1 (resp. Y_2) be the Young diagram that consists of the $(m-1)$ (resp. $(n-1)$) boxes of $X \setminus \{p\}$ on the east (resp. south) side of X. The disjoint sum $Y = Y_1 \sqcup Y_2$ is the diagram of X. (In the terminology of Section 6, Y is the basic diagram of the principal plain diagram X.) The formula (11.2) for this Y is the 'oldest' colored hook formula first appeared in the paper [3] of Greene, Nijenhuis and Wilf.

Notes

[1)] P may be empty.

[2)] For example, in Gaussian elimination to solve a system of linear equations, the elimination process is not uniquely determined.

[3)] We are considering, in the terminology of [2], impartial games.

[4)] This is essentially the same notion as the 'product of directed graphs'.

[5)] The details will appear elsewhere.

[6)] See Note 24) for an example of a plain algorithm that is not finite, and an example of a finite plain algorithm that is not finitely branching.

[7)] More precisely, the points and the arrows of the algorithm $\langle p \rangle_\varphi$ can be explicitly described using only the diagram Y_p and the axioms (P1)-(P4).

[8)] Most results in this section can be generalized, under a suitable formulation, to the finite plain algorithms.

[9)] In this paper, the term 'graph' means, unless otherwise mentioned, a non-oriented simple graph.

[10)] If (P, φ) is connected, then its Dynkin diagram $D(P)_*$ is a tree.

[11)] An error in the Japanese version has been corrected.

[12)] Assume that $E^{(2)}_\varphi(p) = 0$. If $\varphi(p) = \emptyset$, then, by definition, the second player wins. On the other hand, if $\varphi(p) \neq \emptyset$, then, by Part (ii) of Theorem 4.1, we have $E^{(2)}_\varphi(p') > 0$ for any choice $p' \in \varphi(p)$ of the first player, and hence, by Part (i) of Theorem 4.1, the second player can choose a position $p'' \in \varphi(p')$ such that $E^{(2)}_\varphi(p'') = 0$. Continuing this way, the second player always wins. The winning strategy for the first player in the case $E^{(2)}_\varphi(p) > 0$ is similar.

[13)] Some authors call this the Grundy number. See [2, Ch. 11] for the Sprague-Grundy theory.

[14)] See [2, Ch. 6]. The well-known game called Nim [2, Ch. 11] [4, 9.8] is a very special case of the game considered in Section 4.

[15)] If we try to find next good moves at a position p using the method given in Note 12), then we must know for which $q \in \varphi(p)$ the equality $E^{(2)}_\varphi(q) = 0$ holds. For that purpose, it seems, in general, inevitable to calculate the values $E^{(2)}_\varphi(q)$ for all $q \in \varphi(p)$.

[16)] Since $\varphi_{(G,\langle ab \rangle)} = \varphi_2$, the case corresponding to the subgroup $\langle ab \rangle$ can be treated without mentioning Klein's four group.

[17)] An error in the Japanese version has been corrected.

[18)] This notion was introduced by Peterson. See [1] or [16] for its definition.

[19)] The condition (b) can be applied in a wider context than the condition (a).

[20)] Nakayama studied this problem under the influence of R. Brauer's modular representation theory of a finite group.

[21)] Nakayama [14] conjectured that, for a prime number p, two irreducible representations of S_n belong to the same p-block, if and only if the Young diagrams corresponding to them have the same p-core. Although this conjecture was subsequently proved by Brauer and G. de B. Robinson, it is still called Nakayama's Conjecture (in the representation theory of the symmetric groups). See [6, Ch. 6].

[22] A Young diagram with $l-k$ rows ($1 \leq k \leq l$) corresponds to an element of P containing $0, 1, \ldots, k-1$.

[23] Some authors call a board (with stones) like the one in Fig. 7 a 'Maya diagram', which presumably comes from the fact that M. Sato [19] [21] called the game treated in Section 9 (especially the one-line realization of it) 'Maya game'.

[24] The algorithm (P, φ^{-1}) is plain but is not finite. If we generalize the board in Fig. 7 so that the squares of the board corresponds not only to natural numbers but also to transfinite ordinals, then we get a plain algorithm that is finite but is not finitely branching. See Kayada [10], where the 'transfinite Sato-Welter game' is analyzed.

[25] From our point of view, there is no intrinsic difference between a Young diagram and its transpose.

[26] We are still identifying an element of P with the corresponding subset of $\mathbb{N}$.

[27] See Note 21).

[28] See, e.g. [2, Ch. 13].

[29] But, even in Sato's proof [21] (which is quite different from those of Welter and Conway), the Young realization, apparently, does not play a significant role.

[30] The belief is based on: Theorem 2.5 which shows the importance of the notion of a plain diagram in the theory of plain algorithms, and Theorem 4.2 which shows the importance of the same notion in the game-theoretical context.

[31] As already mentioned in Section 4, (9.1) gives the Sprague-Grundy value of Y. Welter [23], Sato [21] and Conway [2] gave another formula for the Sprague-Grundy value of Y (or the corresponding position in the one-line realization) using nim addition. As Sato [19] pointed out, this one is equivalent to (9.1).

[32] In Fig. 12, two connected components of Y' are treated independently; the element a is attached to the box at the north-west corner of each connected component.

References

[1] J. B. Carrell, Vector fields, flag varieties and Schubert calculus, In: Proceedings of the Hyderbad conference on algebraic groups, the Univ. of Hyderbad, Hyderbad, 1989, (eds. S. Ramanan *et al.*), Manoj Prakashan, Madras, (1991), 23-57.

[2] J. H. Conway, On numbers and games, Academic Press, (1976).

[3] C. Greene, A. Nijenhuis and H. S. Wilf, A probabilistic proof of a formula for the number of Young tableaux of a given shape, Advances in Math., **31** (1979), 104-109.

[4] G. H. Hardy and E. M. Wright, An introduction to the theory of numbers, Clarendon Press, (1960).

[5] J. E. Humphreys, Reflection groups and Coxeter groups, Cambridge Univ. Press, (1990).

[6] G. James and A. Kerber, The representation theory of the symmetric groups, Addison-Wesley, (1981).

[7] N. Kawanaka, Sato-Welter game and Kac-Moody Lie algebras, In: Topics in combinatorial representation theory, RIMS Kôkyûroku, **1190** (2001), 95-106. http://www.kurims.kyoto-u.ac.jp/∼kyodo/kokyuroku/contents/pdf/1190-8.pdf

[8] N. Kawanaka, Games, algorithms and representation theory (in Japanese), the abstract of an invited talk presented at the annual meeting 2008, Math. Soc. Japan.

[9] N. Kawanaka, Games with hook structure (in Japanese), In: Proc. the 55th Symp. Algebra (2010), 195-208.

[10] T. Kayada, Some generalizations of Sato's game (in Japanese), In: Combinatorial representation theory and its applications, RIMS Kôkyûroku, **1738** (2011), 24-33.

[11] D. E. Knuth, The art of computer programming, vol. 3 (Sorting and searching), 2nd. ed., Addison-Wesley, (1998).

[12] K. Nakada, Colored hook formula for a generalized Young diagram, Osaka J. of Math., **45** (2008), 1085-1120.

[13] K. Nakada and S. Okamura, An algorithm which generates linear extensions for a generalized Young diagram with uniform probability, In: Proceedings of the 22nd International Conference on Formal Power Series and Algebraic Combinatorics (FPSAC 2010), 933-940.

[14] T. Nakayama, On some modular properties of irreducible representations of a symmetric group, I, II, Jap. J. Math., **17** (1940), 165-184, (1941), 411-423.

[15] S. Okamura, An algorithm which generates, uniform randomly, standard Young tableaux in a generalized sense (in Japanese), Master's thesis, Osaka University, 2003.
[16] R. A. Proctor, Miniscule elements of Weyl groups, the numbers game, and d-complete posets, J. Algebra, **213** (1999), 272-303.
[17] R. A. Proctor, Dynkin diagram classification of λ-miniscule Bruhat lattices and of d-complete posets, J. Algebraic Combin., **9** (1999), 61-94.
[18] M. Sato, On a game (notes by K. Ueno)(in Japanese), In: Proceedings of the 12th symposium of the Algebra Section of the Mathematical Society of Japan, (1968), 123-136.
[19] M. Sato, Mathematical theory of Maya game (notes by H. Enomoto)(in Japanese), Problems of game-playing and puzzle-solving by computer, RIMS Kôkyûroku, **98** (1970), 105-135.
[20] M. Sato, preface, Sugaku no Ayumi **15** (1970), 1-8.
[21] M. Sato, On Maya game (notes by H. Enomoto)(in Japanese), Sugaku no Ayumi **15** (1970), 73-84.
[22] J. R. Stembridge, Miniscule elements of Weyl groups, J. Algebra, **235** (2001), 722-743.
[23] C. P. Welter, The theory of a class of games on a sequence of squares, in terms of the advancing operation in a special group, Indag. Math., **16** (1954), 194-200.

Department of Mathematical Sciences, School of Science and Technology, Kwansei Gakuin University, Sanda, Hyogo 669-1337, Japan