Ph.D. Thesis

ALGEBRAIC PROPERTIES OF PETRI NET LANGUAGES AND CODES

Yoshiyuki Kunimochi

Supervisor: Prof. Pál Dömösi

UNIVERSITY OF DEBRECEN
Doctoral School of Computer Sciences

Debrecen, 2009
# I Background and Aim of the Dissertation

1 Introduction

# II New Results of the Dissertation

3 Automorphism Groups of Nets
   3.1 Transformation Nets
   3.2 Automorphism Groups of Nets
   3.3 Remarks and Further Works

4 Properties of CPN Codes
   4.1 Maximal CPN Codes of the Form $C^n$
   4.2 Maximal CPN Codes of the Form $AB$
   4.3 Constructions of Maximal CPN Codes
   4.4 Rank of CPN Codes
   4.5 Context-sensitiveness of CPN Codes

5 Maximality of CPN Codes
   5.1 Fundamental Properties
   5.1.1 In the case $|P| = 1$ or $|X| = 1$
   5.2 Maximal CPN Codes with two Places
   5.2.1 Without Source Transitions
   5.2.2 With at least one Source Transitions

6 Conclusion

Bibliography

# III Publications

1 Publications related to the dissertation

2 Other Publications
Part I

Background and Aim of the Dissertation
Chapter 1

Introduction

Petri nets are graphical and mathematical modeling tools applicable to many systems. They are promising tools for describing and studying information processing systems that are characterized as being concurrent, asynchronous, distributed, parallel, nondeterministic, and/or stochastic.

However, in many applications modeling by itself is of limited practical use if one cannot analyze the modeled system. As means of gaining a better understanding of the Petri net model, the decidability and computational complexity of typical automata theoretic problems concerning Petri nets have been extensively investigated in the past four decades.

A language over an alphabet $X$ is defined as a subset of the free monoid $X^*$ generated by $X$. A Language of our interest is mainly determined by some procedure that is, computation and derivation, and so on. By applying the concept of a automaton to a Petri net, the Petri net generates a language, called a Petri net language, which is at most context-sensitive. Originally the motivations are to check and validate a system by analyzing the language generated by all possible sequences (words) of system actions, and to automatically synthesis a Petri net that accepts only words of a specific language.

Recently many classes of languages based on Petri nets have been eagerly devised and investigated. For example, some regulated grammars with Petri nets are introduced. Their powers are interesting in the sense that they are often distinct from the classical language classes.

A language $L$ is called a code if it freely generates the submonoid $L^*$ in $X^*$. A prefix code $L$ is a code which no word in $L$ is a proper left factor of any other word in $L$. G.Tanaka defined four types of prefix codes based on Petri nets. He named them an S-type, a D-type, a C-type and a B-type Petri net code, respectively[54].

Chapter 2 plays roles of the introduction for us to the basic notion and of the reference of definitions and notation throughout the literature. These contents is mainly owed to [40], [55]. At first, we introduce the definition of a Petri net and its related concepts and notations. Next, we explain the basic concepts of automata and formal languages. After then, we introduce four types of prefix codes generated by Petri nets and their fundamental properties[54].
Chapter 3 is spent on Petri net structures representing finite groups, which are treated
in [29, 28]. This is a joint work with Professor Genjiro Tanaka and Professor Toshimitsu
Inomata. The problem of automorphism groups of nets is described. We construct a
net, called a transformation net, from a transformation semigroup in a similar way that
we construct an automaton without outputs from a transformation semigroup. It is well
known that for a given group $G$ there exists an automaton such that its automorphism
group is isomorphic to $G$. This fact is proved by using the property that the right regular
representation of a group $G$ commutes with the left regular representation of $G$ as a
permutation group on $G$. An analogous method is applied to prove our main result which
states that for a given finite group $G$ there exists a net $N$ such that its automorphism
group $\text{Aut}(N)$ is isomorphic to $G$. That is, we construct a transformation net which
 corresponds to the right regular representation of a given group $G$ and we show that
$\text{Aut}(N)$ is isomorphic to $G$ by making some arc-weight of the net in certain conditions.

In Chapter 4 we discuss only about C-type Petri net codes (CPN code, for short)
introduced in Chapter 2. This is a joint work[23] with Professor Masami Ito. A CPN
code is the set of all minimal sequences with respect to the prefix order among the firing
sequences through which the state reach from the positive initial marking to a nonpositive
marking. A CPN code of course becomes a prefix code. A CPN code is called a maximal
CPN code if it is a maximal prefix code. In a Petri net which generates a maximal CPN
code, each transition is enable iff every reachable marking is positive. We will investigate
various properties of maximal CPN codes. Moreover, we will prove that a CPN code is a
context-sensitive language in two different ways.

In Chapter 5 we treat the open question raised in Chapter 4. This chapter is completely
my own work. A CPN code generated by some input-ordinary Petri net is called an
input-ordinary CPN code and obviously a maximal CPN code. The problem is whether
$i\text{CPNC} = m\text{CPNC}$ or not, where $i\text{CPNC}$ and $m\text{CPNC}$ means the families of maximal
CPN codes and input-ordinary CPN codes, respectively. It is easily seen that the latter
is a subfamily of the former. But the reverse inclusion is still open in a general Petri net.
So we show that the inclusion is true in restricted cases, i.e., the case that the number of
places is $\leq 2$, and the case that the number of transitions is equal to 1. The general case
still remains open.
Chapter 2
Definitions and Notation

This chapter plays roles of the introduction for us to the basic notion and of the reference of definitions and notation throughout the literature. At first, we introduce the definition of a Petri net and its related concepts and notation. These contents is mainly owed to [40],[55]. Next, we explain the basic concepts of automata and formal languages. After then, we introduce four types of prefix codes generated by Petri nets and their fundamental properties[54].

2.1 Petri net

We introduce the definition of a Petri net and its related concepts and notation.

2.1.1 Definitions and Notation

A Petri net is viewed as a particular kind of directed graph, together with an initial state \( \mu_0 \), called the initial marking. The underlying graph \( N \) of a Petri net is a directed, weighted, bipartite graph consisting of two kinds of nodes, called places and transitions, where arcs are either from a place to a transition or from a transition to a place.

DEFINITION 2.1.1 (Petri net) A Petri net \( PN \) is a 4-tuple, \( PN = (P, T, W, \mu_0) \) where

1. \( P = \{p_1, p_2, \ldots, p_m\} \) is a finite set of places,
2. \( T = \{t_1, t_2, \ldots, t_n\} \) is a finite set of transitions,
3. \( W : E \rightarrow \{0, 1, 2, 3, \ldots\} \), i.e., \( W \in N_0^E \), is a weight function, where \( E = (P \times T) \cup (T \times P) \),
4. \( \mu_0 : P \rightarrow \{0, 1, 2, 3, \ldots\} \), i.e., \( \mu_0 \in N_0^P \), is the initial marking,
5. \( P \cap T = \emptyset \) and \( P \cup T \neq \emptyset \).

When a Petri net structure (net, for short) \( N = (P, T, W) \) without any specific initial marking is denoted by \( N \), a Petri net with a given initial marking \( \mu_0 \) is denoted by \( (N, \mu_0) \).
A Petri net is often given as a 5-tuple \((P, T, F, W, \mu_0)\) adding the set \(F\) of flow relations, i.e., arcs with positive weights: \(F = \{(p, t) \mid W(p, t) > 0\} \cup \{(t, p) \mid W(t, p) > 0\} \subseteq (P \times T) \cup (T \times P)\). Then, a Petri net structure is also given as 4-tuple \(N = (P, T, F, W)\).

Remark A Petri net is often given as a 5-tuple \((P, T, F, W, \mu_0)\) adding the set \(F\) of flow relations, i.e., arcs with positive weights: \(F = \{(p, t) \mid W(p, t) > 0\} \cup \{(t, p) \mid W(t, p) > 0\} \subseteq (P \times T) \cup (T \times P)\). Then, a Petri net structure is also given as 4-tuple \(N = (P, T, F, W)\).

In graphical representation, places are drawn as circles, transitions as bars or boxes. Arcs are labeled with their weights (positive integers), where a \(k\)-weighted arc can be interpreted as the set of \(k\) parallel arcs. Labels for unity weight are usually omitted. A marking (state) assigns a nonnegative integer \(k\) to each place. If a marking assigns a nonnegative integer \(k\) to place \(p\), we say that \(p\) is marked with \(k\) tokens. Pictorially, we put \(k\) black dots (tokens) in place \(p\). A marking is denoted by \(\mu\), an \(n\)-dimensional row vector, where \(n\) is the total number of places. The \(p\)-th component of \(\mu\), denoted by \(\mu(p)\), is the number of tokens in place \(p\).

**Example 2.1.1** Figure 2.1 shows a graphical representation of a Petri net. This Petri net \(PN = (P, T, W, \mu_0)\) represents a process that a bicycle is assembled from one body and two wheels. The places are \(P = \{\text{body}, \text{wheel}, \text{bicycle}\}\) and the transitions are \(T = \{\text{assembly}\}\). Arcs \(f_1 = (\text{body}, \text{assembly})\), \(f_2 = (\text{wheel}, \text{assembly})\) and \(f_3 = (\text{assembly}, \text{bicycle})\) have the weights of 1, 2 and 1, respectively. The other arcs have the weights of 0, and they are not usually drawn in the picture. Note that the weights of \(f_1\) and \(f_3\) is omitted since they are unity. That is, \(W(f_1) = W(f_3) = 1, W(f_2) = 2, W(f) = 0\) for each \(f \in (P \times T) \cup (T \times P) \setminus \{f_1, f_2, f_3\}\).

The initial marking \(\mu_0\) is often denoted by a vector \(\mu_0 = (4, 3, 0)\). The place \(\text{body}\) is marked with three tokens. Then we usually put the number of tokens in a place, instead of black dots (tokens).

In Chapters 4 and 5, we will discuss an input-ordinary Petri net defined in the following definition. The concept of an input-ordinary Petri net is deeply related to the maximality of a Petri net code.

**Definition 2.1.2** (ordinary Petri net) A Petri net \(PN = (P, T, W, \mu_0)\) is called input-ordinary (resp., output-ordinary) if \(W(p, t) \cdot 1\) (resp., \(W(t, p) \cdot 1\)) for each \(p \in P\) and \(t \in T\). A Petri net is called ordinary if it is input-ordinary and output-ordinary.

**Definition 2.1.3** (positive marking) A marking is positive if it is a function from \(P\) to \(N_0 \setminus \{0\}\).

The behavior of many systems can be described in terms of system states and their changes. In order to simulate the dynamic behavior of a system, a state or marking in a Petri net \(PN = (P, T, W, \mu)\) is changed according to the following transition (firing) rule:

1. A transition \(t \in T\) is said to be enabled (under the marking \(\mu\) or under the Petri net \(PN\)) if \(W(p, t) \leq \mu(p)\) for every place \(p \in P\), where \(W(p, t)\) is the weight of the arc.
from \( p \) to \( t \). Then each input place \( p \) of \( t \) is marked with at least \( W(p, t) \) tokens. An enabled transition may or may not fire (depending on whether or not the event actually takes place).

(2) A firing of an enabled transition \( t \) removes \( W(p, t) \) tokens from each input place \( p \) of \( t \), and adds \( W(t, p) \) tokens to each output place \( p \) of \( t \). As a consequence of the firing, the current marking \( \mu \) is replaced with the following new marking \( \mu' \):

\[
\mu'(p) = \mu(p) - W(p, t) + W(t, p) \quad \text{for } \forall p \in P.
\]  

(2.1)

For the equation (2.1), we often use the notation \( \mu [t > \mu' \) (or \((N, \mu) [t > (N, \mu') \), to emphasize the underlying net).

(3) A sequence \( \sigma = t_1 t_2 \ldots t_n \) of transitions is said to be a firing sequence of a Petri net \( PN = (P, T, W, \mu) \) if \( \mu_0 = \mu, \mu_n = \mu' \), and \( \mu_{i-1} [t_i > \mu_i \) for each \( i \) (\( 1 \leq i \leq n \)). Then we often also use the notation \( \mu [\sigma > \mu' \). In particular, a firing sequence \( \sigma \) is said to be positive if all \( \mu_i \) (\( 1 \leq i \leq n \)) are positive.

(4) A marking \( \mu \) is said to be reachable from the initial marking \( \mu_0 \) if there exists a firing sequence \( \sigma \) such that \( \mu_0 [\sigma > \mu \). Then \( \mu \) is said to be reachable from \( \mu_0 \) through \( \sigma \). The set of all possible markings reachable from \( \mu_0 \) in a Petri net \((N, \mu_0) \) is denoted by \( R(N, \mu_0) \) or simply \( R(\mu_0) \). The set of all possible firing sequences from \( \mu_0 \) in a Petri net \((N, \mu_0) \) is denoted by \( L(N, \mu_0) \) or simply \( L(\mu_0) \). The set of all possible positive firing sequences from \( \mu_0 \) in \((N, \mu_0) \) is denoted by \( L_+(N, \mu_0) \) or simply \( L_+(\mu_0) \).

The transition function (or next-state function) \( \delta_{PN} \) of the Petri net \( PN = (N, \mu) \) is defined by

\[
\delta_{PN}(\mu_0, \sigma) = \mu' \quad \text{if } \mu_0 [\sigma > \mu',
\delta_{PN}(\mu_0, \sigma) \text{ is undefined if } \mu' \not\in R(\mu_0).
\]

We may denote \( \delta_{PN} \) by \( \delta \) if no confusion is possible.
EXAMPLE 2.1.2 Consider the Petri net $PN = (P, T, W, \mu_0)$ shown in Figure 2.1. The transition assembly is enabled under the initial marking $\mu_0 = (4, 3, 0)$. Once the transition fires, the marking changes from $\mu_0$ to $\mu_1 = (3, 1, 1)$. Then assembly is not enabled under $\mu_1$ because $W(\text{wheel, assembly}) = 2 \leq \mu(\text{wheel}) = 1$ does not hold. They cannot assemble any more bicycle due to lack of wheels. So the sequences 1 (the empty sequence) and assembly are firing sequences of $PN$ but assembly assembly isn’t a firing sequence. As concerns the next-state function $\delta$ of $PN$, $\delta(\mu_0, 1) = (4, 3, 0)$, $\delta(\mu_0, \text{assembly}) = (3, 1, 1)$, $\delta(\mu_0, \text{assembly assembly})$ is undefined.

For ease of expression, the following notations will be used extensively throughout the literature. Let $(P, T, W, \mu)$ be a Petri net, $p \in P$ be a place, $t \in T$ be a transition and $\sigma$ be a transition sequence. We implicitly assume that some orderings $p_1 \leq p_2 \leq \ldots \leq p_n$ and $t_1 \leq t_2 \leq \ldots \leq t_m$ on $P = \{p_1, p_2, \ldots, p_n\}$ and $T = \{t_1, t_2, \ldots, t_m\}$ are established, respectively.

$\Delta(t)$ is the displacement of $t$, that is, an $n$-dimensional row vector whose $i$-th component is the value of $-W(p_i, t) + W(t, p_i)$. The $i$-th component of $\Delta(t)$ is often denoted by $\Delta(t)(p_i)$. We denote $\Sigma_{i=1}^{k}\Delta(s_i)$ by $\Delta(\sigma)$ where $\sigma = s_1 s_2 \ldots s_k$ ($s_i \in T$). That is, $\Delta(\sigma) = \delta(\mu_0, \sigma) - \delta(\mu_0, 1)$ if $\sigma$ is a firing sequence of the Petri net.

We use the following symbols for a pre-set and a post-set of a place $p \in P$ or a transition $t \in T$:

- $t^\bullet = \{p \in P | W(p, t) > 0\}$ is the set of input places of $t$.
- $\cdot t = \{p \in P | W(t, p) > 0\}$ is the set of output places of $t$.
- $p^\bullet = \{t \in T | W(t, p) > 0\}$ is the set of input transitions of $p$.
- $\cdot p = \{t \in T | W(p, t) > 0\}$ is the set of output transitions of $p$.

A transition $t$ (a) without any output place (i.e., $t^\bullet \neq \emptyset$ and $\cdot t = \emptyset$) (b) with at least one input places and at least one output places (i.e., $t^\bullet \neq \emptyset$ and $\cdot t \neq \emptyset$), (c) without any input place (i.e., $t^\bullet = \emptyset$ and $\cdot t \neq \emptyset$), or (d) without any input place and any output place ($t^\bullet \cup \cdot t = \emptyset$) is called a sink, transform, source or isolated transition, respectively.

Note that a source transition is unconditionally enabled, and that the firing of a sink transition consumes tokens, but does not produce any.

Similarly a place $p$ is called (a) sink, (b) transform, (c) source or (d) isolated place if $p^\bullet \neq \emptyset$ and $p^\bullet = \emptyset$, $p^\bullet \neq \emptyset$ and $p^\bullet = \emptyset$, $p^\bullet = \emptyset$ and $p^\bullet = \emptyset$, or $p^\bullet \cup p^\bullet = \emptyset$, respectively.

2.2 Languages and Codes

We explain terms and notations related to the formal language theory in Section 2.2.1, and codes in Section 2.2.2.
2.2. Languages and Codes

![Figure 2.2: Classification of transitions](image)

2.2.1 Formal Languages

We call a (finite or infinite) set of letters (or symbols) an alphabet. Through the literature we use $X$ as an alphabet only if we don’t specify specially.

A finite sequence of letters in $X$ is called a word (or string) over $X$. The empty word, that is, the word contains no letter, will be denoted by 1. The number of letters occurring in a word $x$ is called the length of $x$ and denoted by $|x|$. In particular, $|1| = 0$ and $|x| = 1 \iff x \in X$.

The set of all words over $X$ attended with a binary associative operation · defined by juxtaposition, sometimes called concatenation;

$$(a_1a_2\ldots a_m) \cdot (b_1b_2\ldots b_n) = a_1a_2\ldots a_mb_1b_2\ldots b_n,$$

forms the semigroup with the identity 1, that is, is the monoid generated by $X$:

$$X^* = 1 \cup X \cup X^2 \cup \ldots \cup X^n \ldots .$$

The base of $X^*$ is obviously the alphabet $X$. Therefore $X^*$ is free, called the free monoid generated by $X$. $X^* = X^* \setminus 1$ is a semigroup, called the free semigroup generated by $X$.

A subset $L$ of $X^*$ is called a language over $X$. A Language of our interest is mainly produced by a mechanical way, that is, computation and derivation, and so on. Here we explain generative grammars which produce languages.

Chomsky grammars

A phase-structure (or type 0) grammar is a quadruple $G = (N, X, S, P)$, where $N, X$ are disjoint alphabets, $S \in N$, $P \subseteq V^*NV^* \times V^*$, for $V = N \cup X$. The elements of $N$ are called nonterminal symbols, those of $X$ are called terminal symbols, $S$ is the start symbol or the axiom, and $P$ is the set of production rules; $(u, v) \in P$ is written in the form $u \to v$.

For $x, y \in V^*$ we write $x \Rightarrow_G y$ iff $x = x_1ux_2, y = x_1vx_2$, for some $x_1, x_2 \in V^*$ and $u \to v \in P$. If $G$ is understood, we write $x \Rightarrow y$. The reflective and transitive closure
2.2. Languages and Codes

of the relation $\Rightarrow$ is denoted by $\Rightarrow^*$. The language generated by $G$ is $\{ x \in X^* | S \Rightarrow^* x \}$, denoted by $L(G)$.

A phase-structure grammar $G = (N, X, S, P)$ is called:

- **context-sensitive** (or type 1) if each $u \to v \in P$ has $u = u_1Au_2, v = u_1xu_2$ for $u_1, u_2 \in V^*, A \in N, x \in V^+$.
- **context-free** (or type 2) if each production $u \to v \in P$ has $u \in N$.
- **linear** if each production $u \to v \in P$ has $u \in N$ and $v \in X^*NX^*$.
- **left-linear** (or type 3) if each rule $u \to v \in P$ has $u \in N$ and $v \in X \cup XN \cup \{ \lambda \}$.
- regular if each rule $u \to v \in P$ has $u \in N$ and $v \in X^* XN^*$.

In context-sensitive grammars, a production $S \to \lambda$ is allowed, providing $S$ does not appear in the right-hand members of rules in $P$.

A language generated by phase-structure (resp., context-sensitive, context-free, regular) grammar is called a phase-structure (resp., context-sensitive, context-free, regular) language and can be accepted by a Turing machine (resp., a linear-bounded Turing machine, a pushdown automaton, a finite automaton).

We denote by $RE, CS, CF, LIN, REG$ the family of languages generated by phase-structure, context-sensitive, context-free, linear and regular grammars, respectively.

The well-known Chomsky hierarchy, that is, the following strict inclusion, holds: $REG \subset LIN \subset CF \subset CS \subset RE$.

### 2.2.2 Codes

Let $X^*$ be the free monoid generated by an alphabet $X$. A **code** is the base of a free submonoid $M$ in $X^*$, and conversely it freely generates the submonoid $M$ in $X^*$. It is formally defined as follows:

A nonempty language $C$ is a code if for any two integers $n, m \geq 1$ and any words $u_1, u_2, \ldots, u_n, v_1, v_2, \ldots, v_m \in C$,

$$u_1u_2\cdots u_n = v_1v_2\cdots v_m$$

implies

$$n = m \text{ and } u_i = v_i \text{ for } i = 1, \ldots, n.$$

If for two words $w, u \in X^*$ there exists some word $v \in X^*$ with $w = uv$ (resp., $w = vu$), then $u$ is called a **prefix** (resp., **suffix**) or a **left factor** (resp., **right factor**) of $w$, and denoted by $u \unlhd_p w$ (resp., $u \unlhd_s w$). A prefix $u$ (resp., a suffix $u$) of $w$ is called proper if $u \neq w$, and denoted by $u <_p w$ (resp., $u <_s w$). A word $u$ is a **subword** of a word $w$ if there exist words $v_1$ and $v_2$ (possibly empty) such that $w = v_1uv_2$.

A language $L$ becomes a code, called a **prefix code**, if $u, uv \in L$ implies $v = 1$ (equivalently $L \cap LX^+ = \emptyset$). A **suffix code** is defined left-right dually. A prefix and suffix code is
called a bifix code. A language \( L \) becomes a bifix code, called a infix code if \( u, v_1uv_2 \in L \) implies \( v_1 = v_2 = 1 \). A nonempty subset \( U \) of \( X^n = \{ w \mid |w| = n \} \) for some positive integer \( n \) is an infix code, called a uniform code. Especially the uniform code \( U = X^n \) is called a full uniform code. These codes have the following strict implication:

\[
\text{full uniform } \Rightarrow \text{uniform } \Rightarrow \text{infix } \Rightarrow \text{bifix } \Rightarrow \text{prefix/suffix}.\]

A code (resp., prefix code) \( C \subseteq X^+ \) is maximal (resp., maximal prefix) in \( X \) if \( C \) is not included by any other code (resp., prefix code) over \( X \).

Remark A maximal and prefix code is clearly a maximal prefix code because it is not included in any code by the maximality. But a maximal prefix code is a prefix code, but is not necessarily a maximal code.[2]

### 2.3 Petri Net Codes

G. Tanaka defined four types of prefix codes, called S-type, D-type, C-type and B-type Petri net codes, respectively, based on Petri nets[54]. Note that these codes is a Petri net language whose element is a firing sequence itself and labelling function is the identity mapping. Otherwise a obtained language cannot form a code. In this section we explain their definitions and summarize fundamental properties of these codes.

#### DEFINITION 2.3.1

Let \( PN = (N, \mu_0) = (P, X, W, \mu_0) \) be a Petri net. The set

\[
\text{Stab}(\mu_0) = \{ w \mid w \in L(\mu_0) \text{ and } \delta(\mu_0, w) = \mu_0 \}
\]

forms a free submonoid of \( X^* \). The base of \( \text{Stab}(\mu_0) \), that is

\[
(\text{Stab}(\mu_0) \setminus \{1\}) \setminus (\text{Stab}(\mu_0) \setminus \{1\})^2,
\]

is said to be an S-type Petri net code (SPN code or SPNC, for short) if it is not empty, and denoted by \( S(PN) \) or \( S(N, \mu_0) \).

Since \( S(PN, \mu_0)X^+ \cap S(PN, \mu_0) = \emptyset \), \( S(PN, \mu_0) \) is a prefix code over \( X \). The following set \( D(PN, \mu_0) \) is a subset of \( S(PN, \mu_0) \), so it is also a prefix codes.

#### DEFINITION 2.3.2

Let \( PN = (N, \mu_0) = (P, X, W, \mu_0) \) be a Petri net with a positive marking \( \mu_0 \). The set of all positive firing sequences of \( S(PN, \mu_0) \) is said to be a D-type Petri net code (DPN code or DPNC, for short), and denoted by \( D(PN) \) or \( D(N, \mu_0) \).

#### EXAMPLE 2.3.1

Let \( PN = (P, T, W, \mu_0) \) be a Petri net where \( \mu_0 = (2, 4) \) shown in Figure 2.3. Observing the reachability graph of \( PN \), we obviously have \( \text{Stab}(\mu_0) = \{a^3b, a^2ba\}^* \) and \( S(PN) = \{a^3b, a^2ba\} \). An element in \( S(PN) \) is a nonempty minimal word in \( \text{Stab}(\mu_0) \) with respect to the prefix order \( \leq_p \). Since an element in \( D(PN) \) is a positive firing sequence \( \sigma \) in \( S(PN) \) which for any prefix \( u \) of \( \sigma \) \( \delta(\mu_0, u) \) must be positive, \( a^3b \) is in \( D(PN) \) but \( a^2ba \) is not. Hence, \( D(PN) = \{a^3b\} \).
DEFINITION 2.3.3 Let $PN = (N, \mu_0) = (P, X, W, \mu_0)$ be a Petri net with a positive marking $\mu_0$. By $C(PN)$ or $C(N, \mu_0)$ denoted the set of all sequences $w \in L(\mu_0)$ satisfying the following conditions:

1. $\delta(\mu_0, w)$ is not a positive marking, i.e., $w \in L(\mu_0) \setminus L_+(\mu_0)$.
2. $\delta(\mu_0, v)$ is a positive marking for any proper prefix $v$ of $w$ i.e., $v \in L_+(\mu_0)$.

DEFINITION 2.3.4 Let $PN = (N, \mu_0) = (P, X, W, \mu_0)$ be a Petri net with a positive marking $\mu_0$. By $B(PN)$ or $B(N, \mu_0)$ denoted the set of all sequences $w \in C(PN, \mu_0)$ satisfying $\delta(\mu_0, v) \neq \mu_0$ for any prefix $v(\neq 1)$ of $w$.

By the definition 2.3.3, $C(PN)$ is obviously a prefix code over $X$ if it is not empty. Since the set $B(PN)$ is a subset of the prefix code $C(PN)$, so that $B(PN)$ is also a prefix code over $X$. Then $C(PN)$ and $B(PN)$ are said to be a $C$-type Petri net code (CPN code or CPNC, for short) and $B$-type Petri net code (BPN code or BPNC, for short), respectively.

The following proposition shows the fundamental relationship among a BPN code, a CPN code and a DPN code.

PROPOSITION 2.3.1 [54]Let $PN = (P, X, W, \mu_0)$ be a Petri net with a positive marking $\mu_0$. Then

$$C(PN) = D(PN)^*B(PN).$$
EXAMPLE 2.3.2 Again let $PN = (P, T, W, \mu_0)$ be a Petri net where $\mu_0 = (2, 4)$ shown in Figure 2.3. $C(PN, \mu_0) = \{a^3b, a^2ba\}^*\{a^4, a^2b\}$. Noting that $\delta(\mu_0, a^3b) = \delta(\mu_0, a^2ba) = \mu_0$, Only elements $a^4$ and $a^2b$ in $C(PN)$ are in $B(PN)$, the others are not, where $\delta$ is the next-state function of the Petri net. 

The family of SPN codes (resp., DPN codes, CPN codes, BPN codes) is denoted by SPNC (resp., DPNC, CPNC, BPNC). The following inclusions are obvious.

$$DPNC \subseteq SPNC \quad \text{and} \quad BPNC \subseteq CPNC.$$ 

We will discuss the maximality of CPN codes in the following two chapters. Here we prepare the related terminologies and notations. A CPN code is said to be a maximal C-type Petri net code (maximal CPN code or mCPNC, for short) if it is a maximal prefix code. The family of all maximal CPN codes is denoted by mCPNC.

A CPN code $C$ is said to be a input-ordinary C-type Petri net code (input-ordinary CPN code or iCPNC, for short) if $C = C(PN)$ for some input-ordinary Petri net $PN$. The family of all input-ordinary CPN codes is denoted by iCPNC.

Since an input-ordinary CPN code is clearly an maximal CPN code, we have the inclusion relation $iCPNC \subseteq mCPNC$. The following problem remains open.

Problem] $mCPNC \subseteq iCPNC$ ?

Since it is too difficult to solve this problem in general Petri nets, in Chapter 5 we prove that the problem is solved affirmatively in some restricted Petri nets.

The notion of maximality of a CPN code is very important in relation to liveness in the following sense. Let $C(PN) \neq \emptyset$, $PN = (P, X, W, \mu_0)$ with a positive marking $\mu_0$ be a maximal CPN code. If $\mu \in L_+(\mu_0)$, that is, $\mu$ is reachable from $\mu_0$ by a positive firing sequence, then every transition is enable at the marking $\mu$. In other words, the assumption is equivalent to $u \in C(PN)(X^+)^{-1}$, that is, $u$ is a proper prefix of $C(PN)$. 
Part II

New Results of the Dissertation
Chapter 3

Automorphism Groups of Nets

In this chapter we discuss the problem of automorphism groups of nets [29]. We construct a net called a transformation net from a transformation semigroup in a similar way to how we construct an automaton without outputs from a transformation semigroup[10]. It is well known that for a given group $G$ there exists an automaton such that its automorphism group is isomorphic to $G$. This fact is proved by using the property that the right regular representation of a group $G$ commutes with the left regular representation of $G$ as a permutation group on $G$. An analogous method is applied to prove our main result which states that We slightly touch on the generalization of the right regular representation of a group. That is, we construct a transformation net which corresponds to the right regular representation of a given group $G$ and we show that $\text{Aut}(N)$ is isomorphic to $G$ by making some arc-weight of the net in certain conditions.

3.1 Transformation Nets

In this section, we define an automorphism of a net and a net called a transformation net. We represent a finite group by using some transformation net.

DEFINITION 3.1.1 A net is a triple $(P, T, W)$ satisfying the following conditions (i) and (ii).

(i) $P$ and $T$ are finite nonempty sets with $P \cap T \neq \emptyset$. An element of $P$ (resp., $T$) is called a place (resp. a transition).

(ii) $W: (P \times T) \cup (T \times P) \rightarrow \{0, 1, 2, \ldots\}$ is called a weight function. Moreover, $a \in F$ iff $W(a) > 0$.

The subset $F$ of $(P \times T) \cup (T \times P)$ is called the flow relation of a net $(P, T, W)$ if $F = \{(p, t) \in P \times T | W(p, t) > 0\} \cup \{(t, p) \in T \times P | W(t, p) > 0\}$. An element of $F$ is called an arc.

DEFINITION 3.1.2 Let $N_1 = (P_1, T_1, W_1)$ and $N_2 = (P_2, T_2, W_2)$ be nets, and let $\alpha : P_1 \rightarrow P_2$ and $\beta : T_1 \rightarrow T_2$ be bijections. We define the mapping $(\alpha, \beta) : (P_1 \times T_1) \cup (T_1 \times P_1) \rightarrow (P_2 \times T_2) \cup (T_2 \times P_2)$ by
DEFINITION 3.1.3 A net $N = (P, T, W)$ is said to be transformation type if

(i) $P$ is the union of nonempty sets $Q$ and $S$ with $Q \cap S = \emptyset$. We call the element of $Q$ (resp., $S$) an inner (resp., source) place.

(ii) For each $t \in T$,
$$S \cap t \bullet = \emptyset \quad \text{and} \quad |t \cap Q| = |t \cap S| = |Q \cap t \bullet| = 1,$$
where $|X|$ is the cardinality of a set $X$.

(iii) For each $(q, s) \in Q \times S$, there exists a unique $t \in T$ such that $t \bullet = \{q, s\}$.

After this, a transformation type net is called transformation net simply.

Let $S$ be a transformation semigroup on a set $Q$, then we can define a transformation net $N = (Q \cup S, Q \times S, W)$, where $F$ is the flow relation of it and
$$F = \{((p, (p, s)))|p \in Q, s \in S\} \cup \{(s, (p, s))|p \in Q, s \in S\} \cup \{(p, s)|p \in Q, s \in S\}.$$

Conversely, let $N = (Q \cup S, T, W)$ be a transformation net, and let $S \subseteq T$ be a source place. For each $q \in Q$ there exists a unique $t \in T$ such that $(q, t), (s, t) \in F$. Therefore we can define a transformation $s'$ on $Q$ by $s' : q \mapsto p$, where $t \bullet = \{p\}$.

Let $G$ be a group. For any $y \in G$, $y'$ denotes the transformation defined by $y' : G \rightarrow G : x \mapsto xy$, and by $G'$ we denote the group $\{y'|y \in G\}$ with the multiplication of $y'$, $z' \in G'$ defined by $y' \cdot z'(x) = z'(y'(x))$. It is obvious that $G'$ is isomorphic to $G$.

DEFINITION 3.1.4 Let $G$ be a finite group. A net $N = (G \cup G', G \times G', W)$ is called a transformation net of $G$ if its flow relation $F$ satisfies
$$F = \{(x, (x, y'))|x \in G, y' \in G'\} \cup \{(y', (x, y'))|x \in G, y' \in G'\} \cup \{((x, y'), xy)|x, y \in G, y' \in G'\}.$$
3.2 Automorphism Groups of Nets

In this section we construct a transformation net which corresponds to the right regular representation of a given group $G$ and we show that for an arbitrarily finite group $G$ there exists a net $N$ such that $\text{Aut}(N)$ is isomorphic to $G$ by introducing appropriate weights of arcs in the net.

First of all, it is easily verified that a set of all the automorphisms of a net $N = (P, T, W)$ is a group with the identity $(1_P, 1_T)$, where $1_P$ and $1_T$ are identity mappings of $P$ and $T$, respectively.

**Theorem 3.2.1** Let $G$ be a finite group and let $N_1 = (G \cup G', G \times G', W_1)$ be a transformation net of $G$, where the weight function $W_1$ is defined as follows:

- $W_1$: For each $x \in G$ and each $y' \in G'$, $W_1(x, (x, y')) = W_1((x, y'), xy) = 1$ and $W_1(y', (x, y')) = \rho(y')$, where $\rho : G' \rightarrow \{2, 3, \ldots\}$ is injective.

Then the automorphism group $\text{Aut}(N_1)$ of $N_1$ is isomorphic to $G$.

The following Theorem 3.2.2 is another expression of Theorem 3.2.1.

**Theorem 3.2.2** For a given finite group $G$, there exists a net $N$ such that its automorphism group $\text{Aut}(N)$ is isomorphic to $G$.

Since an automorphism group $\text{Aut}(G)$ of a finite group $G$ is also a finite group, we have the following corollary by Theorem 3.2.2. However, in the proof of the following corollary we construct another net in a different way shown in the proof of Theorem 3.2.1.
3.3. Remarks and Further Works

The construction method of the net in the proof of COROLLARY 3.2.1 is effective for the case that $G$ is an automorphism group of some group $H$ and the order of $H$ is smaller than that of $G$, because we can construct the transformation net $N$ of $H$ which has fewer number of places and transitions than that of the transformation net of $G$.

COROLLARY 3.2.1 For a finite group $G$ there exists a net $N$ such that its automorphism group $\text{Aut}(N)$ is isomorphic to $\text{Aut}(G)$. □

3.3 Remarks and Further Works

The notion of the automorphism group $\text{Aut}(N)$ of a Petri net structure $N$ is newly introduced. We show the main theorem that for a given finite group $G$ there exists a Petri net structure $N$, called a transformation net, such that $\text{Aut}(N)$ is isomorphic to $G$. The structure $N$ corresponds to the right regular representation of $G$.

A transformation net is only a Petri net structure without marking. We can consider a behavior of a Petri net by adding a marking to it. I have had an equivalent condition to the reachability problem with respect to a transformation net $N = (G \cup G', G \times G', W)$, where $G$ is the residue group of order $n$ [27].

Let $\mu$ and $\lambda$ be markings with $\lambda(g^0) = 0$ for any $g^0 \in G$ (we may assume this condition without loss of generality). Then if $\mu$ is reachable from $\lambda$, then

\begin{align*}
(1) & \quad \sum_{g \in G} \mu(g) = \sum_{g \in G} \lambda(g), \\
(2) & \quad \sum_{g \in G} g \cdot (\mu(g) + \mu(g')) \equiv \sum_{g \in G} g \cdot \lambda(g) \pmod{n}.
\end{align*}

Though this is just a necessary condition for the reachability. We can compute and check the conditions (1) and (2) in $O(n)$ time, where $n = |G|$.

The remaining problem is the reverse implication, that is, whether both (1) and (2) imply that $\mu$ is reachable from $\lambda$ or not.
Chapter 4

Properties of CPN Codes

In this chapter, we consider the language over an alphabet $X$ generated by a given Petri net with a positive initial marking, called a CPN code. This code becomes a prefix code over $X$. We are interested in CPN codes which are maximal prefix codes, called maximal CPN codes over $X$. We will investigate various properties of maximal CPN codes. Moreover, we will prove that a CPN code is a context-sensitive language in two different ways.

4.1 Maximal CPN Codes of the Form $C^n$

Let $D = (P, X, W, \mu_0)$ be a Petri net with an initial marking $\mu_0$ where $P$ is the set of places, $X$ is the set of transitions, $W$ is the weight function and $\mu_0 \in N^P$ is a positive marking, i.e. $\mu_0(p) > 0$ for any $p \in P$. Notice that $\mu_0(p)$ is meant the number of tokens at $p$ of the marking $\mu_0$.

Let $\delta$ be the next-state function of $D$. A language $C$ is called a CPN code over $X$ generated by $D$ and denoted by $C = C(D)$ if $C = \{u \in X^+ \mid \exists p \in P, \delta(\mu_0, u)(p) = 0, \forall q \in P, \delta(\mu_0, u)(q) \geq 0, \text{ and } \forall q' \in P, \delta(\mu_0, u')(q') > 0 \}$ for $u' \in P_r(u) \setminus \{u\}$ where $P_r(u)$ is the set of all prefixes of $u$. Then it is obvious that $C = C(D)$ is a prefix code over $X$ if $C = C(D) \neq \emptyset$. Notice that CPN codes were introduced in [54]. If $C$ is a maximal prefix code, then $C$ is called an maximal CPN codes over $X$. Now let $u = a_1a_2 \ldots a_r \in X^+$ where $a_i \in X$. Then, for any $p \in P$, by $p(u)$ we denote $-\Delta(u)(p)$, that is, the negative value at the place $p$ of the displacement $\Delta(u)$ of $u$, which is the amount of consumed tokens at $p$ by the firing sequence $u$.

**Lemma 4.1.1** Let $C = C(D)$ be a finite maximal CPN code where $D = (P, X, W, \mu_0)$. By $t_p$ we denote $\mu_0(p)$ for any $p \in P$. For any $u, v \in C$, if there exists a $p \in P$ such that $t_p = p(u) = p(v)$, then $C$ is a full uniform code over $X$, i.e. $C = X^n$ for some $n, n \geq 1$.

**Lemma 4.1.2** Let $A, B$ be finite maximal prefix codes over $X$. If $AB$ is a maximal CPN code over $X$, then $B$ is a maximal CPN code over $X$.

**Corollary 4.1.1** Let $C^n$ be a finite maximal CPN code over $X$ for some $n, n \geq 2$. Then $C^k$ is a maximal CPN code over $X$ for any $k, 1 \leq k \leq n$.
4.2. Maximal CPN Codes of the Form $AB$

Now we provide a fundamental result.

**PROPOSITION 4.1.1** Let $C^n$ be a finite maximal CPN code over $X$ for some $n$, $n \geq 2$. Then $C$ is a full uniform code over $X$. 

**COROLLARY 4.1.2** The property being a maximal CPN code over $X$ is not preserved under concatenation.

**REMARK 4.1.1** We can prove the above corollary in a different way. Let $X = \{a, b\}$, let $A = \{a, ba, bb\}$ and let $B = \{b, ab, aa\}$. Then $ab, aaa, bbb \in AB$ and $|aaa|, |bbb| > |ab|$. By the following lemma, $AB$ is not a maximal CPN code over $\{a, b\}$.

**LEMMA 4.1.3** Let $C \subseteq X^+$ be a maximal CPN code over $X$. Then there exists $a \in X$ such that $a^{|u|u \in C^+} \in C$.

**REMARK 4.1.2** If $C$ is an infinite maximal CPN code over $X$, then PROPOSITION 4.1.1 does not hold true. For instance, let $X = \{a, b\}$ and let $C = b^*a$. Then both $C$ and $C^2 = b^*ab^*a$ are maximal CPN codes over $X$.

4.2 Maximal CPN Codes of the Form $AB$

We can generalize PROPOSITION 4.1.1 to a maximal CPN code of the form $AB$ as follows:

**PROPOSITION 4.2.1** Let $A, B$ be finite maximal prefix codes over $X$. If $AB$ is a maximal CPN code over $X$, then $A$ and $B$ are full uniform codes over $X$.

4.3 Constructions of Maximal CPN Codes

In this section, we provide two construction methods of maximal CPN codes.

**DEFINITION 4.3.1** Let $A, B \subseteq X^+$. Then by $A \oplus B$ we denote the language $(\cup_{b \in X} \{(P_r(A) \circ Bb^{-1})b\}) \cup (\cup_{a \in X} \{(P_r(B) \circ Aa^{-1})a\})$ where $\circ$ is meant the shuffle operation and $Ca^{-1} = \{u \in X^+|ua \in C\}$ for $C \subseteq X^+$ and $a \in X$.

**PROPOSITION 4.3.1** Let $X = X_1 \cup X_2$ where $X_1, X_2 \neq \emptyset, X_1 \cap X_2 = \emptyset$. If $A \subseteq X_1^+$ is a maximal CPN code over $X_1$ and $B \subseteq X_2^+$ is a maximal CPN code over $X_2$, then $A \oplus B$ is a maximal CPN code over $X$.

**EXAMPLE 4.3.1** Let $X = \{a, b\}$. Consider $A = \{a\}$ and $B = \{bb\}$. Then both $A$ and $B$ are maximal CPN codes over $\{a\}$ and $\{b\}$, respectively. Hence $A \oplus B = \{a, ba, bb\}$ is a maximal CPN codes over $X$. 


4.3. Constructions of Maximal CPN Codes

**PROPOSITION 4.3.2** Let \(A, B \subseteq X^+\) be finite maximal CPN codes over \(X\). Then \(A \oplus B\) is a maximal CPN codes over \(X\) if and only if \(A = B = X\). \(\square\)

**REMARK 4.3.1** For the class of infinite maximal CPN codes over \(X\), the situation is different. For instance, let \(X = \{a, b\}\) and let \(A = B = b^*a\). Then \(A \oplus B = b^*a\) and \(A, B\) and \(A \oplus B\) are maximal CPN codes over \(X\). \(\square\)

**PROPOSITION 4.3.3** Let \(A, B \subseteq X^+\) be maximal CPN codes over \(X\). Then there exist an alphabet \(Y\), a maximal CPN code \(D \subseteq Y^+\) over \(Y\), a \(\lambda\)-free homomorphism \(h\) of \(Y \rightarrow X\) such that \(A \cap B = h(D)\). \(\square\)

**THEOREM 4.3.1** The property being a maximal CPN code over \(X\) is not preserved under \(\lambda\)-free homomorphism. \(\square\)

**LEMMA 4.3.1** Let \(C \subseteq X^+\) be a maximal CPN code over \(X\) and let \(a, b \in X\). If \(bbaa \in C, \) then \(baba \in C\). \(\square\)

**REMARK 4.3.2** By the above lemma, a maximal prefix code over \(X\) having the property in LEMMA 4.3.1 cannot be necessarily realized by a Petri net. For instance, let \(X = \{a, b\}\) and let \(C = \{a, ba, bbba, bbb\}\). Then \(C\) is a maximal prefix code over \(X\). However, by LEMMA 4.3.1, it is not a maximal CPN code over \(X\). \(\square\)

Now we introduce another method to construct maximal CPN codes.

**DEFINITION 4.3.2** Let \(A \subseteq X^+\). By \(m(A)\), we denote the language \(\{v \in A | \forall u, v \in A, \forall x \in X^+, v = ux \Rightarrow x = 1\}\). Obviously, \(m(A)\) is a prefix code over \(X\). Let \(A, B \subseteq X^+\). By \(A \otimes B\), we denote the language \(m(A \cup B)\). \(\square\)

**PROPOSITION 4.3.4** Let \(A, B\) be maximal CPN codes over \(X\). Then, \(A \otimes B\) is a maximal CPN code over \(X\). \(\square\)

**EXAMPLE 4.3.2** It is obvious that \(a^*b\) and \((a \cup b)^3\) are maximal CPN codes over \(\{a, b\}\). Hence \(a^*b \otimes (a \cup b)^3 = \{b, ab, aab, aab\}\) is a maximal CPN code over \(\{a, b\}\). \(\square\)

**REMARK 4.3.3** PROPOSITION 4.3.4 does not hold for the class of CPN codes over \(X\). The reason is the following: Suppose that \(A \otimes B\) is a CPN code over \(X\) for any two CPN codes \(A\) and \(B\) over \(X\). Then we can show that, for a given finite CPN code \(A\) over \(X\), there exists a finite maximal CPN code \(B\) over \(X\) such that \(A \subseteq B\) as follows. Let \(A \subseteq X^+\) be a finite CPN code over \(X\) which is not a maximal CPN code. Let \(n = \max \{|u||u \in A\}\). Consider \(X^n\) which is a maximal CPN code over \(X\). By assumption, \(A \otimes X^n\) becomes a CPN code (in fact, a maximal CPN code) over \(X\). By the definition of the operation \(\otimes\), it can be also proved that \(A \subseteq A \otimes X^n\). However, as the following example shows, there exists a finite CPN code \(A\) over \(X\) such that there exists no maximal CPN code \(B\) over \(X\) with \(A \subseteq B\). Hence, PROPOSITION 4.3.4 does not hold for the class of all CPN codes over \(X\). \(\square\)
EXAMPLE 4.3.3 Consider the language $A = \{ab, aaba, aaa\} \subseteq \{a, b\}^+$. Then this language becomes a CPN code over $\{a, b\}$ (see Fig. 4.1). Moreover, it can be proved that there is no maximal CPN code $B$ over $\{a, b\}$ with $A \subseteq B$ as follows: Suppose $B \subseteq \{a, b\}^+$ is a maximal CPN code with $A \subseteq B$ over $X$. By LEMMA 4.1.3, $b \in B$ or $b^2 \in B$. Let $b^i \in B$ where $i = 1$ or 2. Let $t_p = p(ab)$ where $p \in P$ and $P$ is the set of places of the Petri net which recognizes $B$. If $p(a) < 0$. Then $p(b) > t_p$ and hence $p(b^i) > t_p$. This contradicts the fact $b^i \in B$. If $p(a) > 0$, then $p(aaba) = p(ab) + 2p(a) > t_p$. This contradicts the fact $aaba \in B$ as well. Hence $p(a) = 0$ and $p(aab) = t_p$. However, since $aab$ is a prefix of $aaba \in B$, $p(aab) < t_p$. This yields a contradiction again. Therefore, there is no maximal CPN code $B \subseteq \{a, b\}^+$ with $A \subseteq B$. 

![Figure 4.1: Petri net $D$ with $C(D) = \{ab, aaba, aaa\}$](figure)

REMARK 4.3.4 The set of all maximal CPN codes over $X$ forms a semigroup under $\otimes$. Moreover, the operation $\otimes$ has the following properties:

1. $A \otimes B = B \otimes A$,
2. $A \otimes A = A$,
3. $A \otimes X = X$.

Consequently, the set of all maximal CPN codes over $X$ forms a commutative band with zero under $\otimes$ (for bands, see [5]).

4.4 Rank of CPN Codes

In this section, we will consider the rank and related decomposition of CPN codes.

DEFINITION 4.4.1 Let $A \subseteq X^+$ be a CPN code over $X$. By $r(A)$ we denote the value $\min\{|P| \mid D = (P, X, W, \mu_0), C(D) = A\}$.

REMARK 4.4.1 Let $A \subseteq X^+$ be a finite maximal CPN code over $X$. Then $r(A) \leq |A|$. The proof can be done as follows: Let $D = (P, X, W, \mu_0)$ be a Petri net with a positive initial marking $\mu_0$ such that $C(D) = A$, and $\delta$ the next-state function of $D$. Let $P' = \{p_u \in P \mid u \in A, p_u(u) = \delta(\mu_0, u)\} \subseteq P$. The transition function $\delta'$ can be defined as $\delta'(\mu|_{P''}, a) = \delta(\mu, a)|_{P''}$ where $a \in X$. Then $A = C(D')$ and it is obvious that $r(A) \leq |A|$. However, in general this inequality does not hold for a CPN code over $X$ as the following example shows. In Figure 4.2, $C(D) = \{aba\}$ but $r(\{aba\}) \neq 1$ because $aba \in A$ if and only if $baa \in A$ for any CPN code with $r(A) = 1$. 

![Figure 4.2: Petri net $D'$ with $C(D') = \{aba\}$](figure)
4.4. Rank of CPN Codes

Now let $A, B \subseteq X^+$ be maximal CPN codes over $X$. Then it is easy to see that $|A \otimes B| \leq \max\{|A|, |B|\}$. Moreover, if $A$ and $B$ are finite, then $r(A \otimes B) \leq r(A) + r(B)$.

We define three language classes as follows: $\text{CPN} = \{A \subseteq X^+ \mid A \text{ is a CPN code over } X\}$, $\text{mCPN} = \{A \subseteq X^+ \mid A \text{ is an maximal CPN code over } X\}$, $\text{iCPN} = \{A \subseteq X^+ \mid A \text{ is a CPN code over } X, \exists D = (P, X, W, \mu_0), \forall p \in P, \forall a \in X, W(p, a) \leq 1, C(D) = A\}$. Then it is obvious that we have the following inclusion relations: $\text{iCPN} \subseteq \text{mCPN} \subseteq \text{CPN}$.

It is also obvious that $\text{mCPN} \neq \text{CPN}$.

Problem 4.1 Does $\text{mCPN} = \text{iCPN}$ hold?

\begin{proposition}
Let $A \in \text{mCPN}$. Then there exist a positive integer $k \geq 1$ and $A_1, A_2, \ldots, A_k \in \text{CPN}$ such that $r(A_i) = 1, i = 1, 2, \ldots, k$ and $A = A_1 \otimes A_2 \otimes \cdots \otimes A_k$. Moreover, in the above, if $A \in \text{iCPN}$, then $A_1, A_2, \ldots, A_k$ are in $\text{iCPN}$ and context-free.
\end{proposition}

\begin{problem}
In the above proposition, can we take $r(A)$ as $k$ if $A \in \text{iCPN}$?
\end{problem}

\begin{proposition}
Let $A \subseteq X^+$ be a finite maximal CPN code with $r(A) = 1$ over $X$. Then $A$ is a full uniform code over $X$.
\end{proposition}

\begin{proposition}
Let $A \subseteq X^+$ be a maximal CPN code with $r(A) = 1$ over $X$ and let $k$ be a positive integer. Then $A^k$ is a maximal CPN code with $r(A^k) = 1$ over $X$.
\end{proposition}

\begin{proposition}
Let $A \in \text{iCPN}$. Then, by PROPOSITION 4.4.3, there exist $A_1, A_2, \ldots, A_k \in \text{iCPN}$ such that $r(A_i) = 1, i = 1, 2, \ldots, k$ and $A = A_1 \otimes A_2 \otimes \cdots \otimes A_k$. Let $n_1, n_2, \ldots, n_k$ be positive integers. Then $A_1^{n_1} \otimes A_2^{n_2} \otimes \cdots \otimes A_k^{n_k} \in \text{iCPN}$.
\end{proposition}
4.5 Context-sensitivity of CPN Codes

Consider the Petri net $D = (P, X, W, \mu_0)$ depicted below. Then $C(D) \cap a^+b^+c^+ = \cup_{n \geq 1} \{a^nb^nc^{i+1} | 1 \leq i \leq n\}$ is not context-free. Hence $C(D)$ is not context-free. Therefore, the class of all CPN codes over an alphabet $X$ is not necessary included in the class of all context-free languages over $X$. However, in this section, we will prove the context-sensitivity of CPN codes.

![Petri net diagram]

**Figure 4.3:** Petri net which generates a non-context-free language

**THEOREM 4.5.1** Let $C \subseteq X^+$ be a CPN code over $X$. Then $C$ is a context-sensitive language over $X$. \(\square\)

Before giving the second proof, we provide a few notations. Let $\mu_1, \mu_2, \ldots, \mu_r$ and $\mu$ be markings of a Petri net. Then $\mu = \mu_1 + \mu_2 + \ldots + \mu_r$ if $\mu(p) = \mu_1(p) + \mu_2(p) + \ldots + \mu_r(p)$ for any $p \in P$. Now let $D = (P, X, W, \mu_0)$ be a Petri net with a positive initial marking $\mu_0$. Let $N_D = \max\{W(p,a), W(b,q) | a, b \in X, p, q \in P\}$ and let $M_D = \max\{\mu_0(p) | p \in P\}$. By $\Omega_D$ we denote the set of markings $\{\mu | \forall p \in P, \mu(p) \leq M_D + 3N_D\}$. Notice that $\Omega_D$ is a finite set.

**(Sketch of Proof)** Let $D = (P, X, W, \mu_0)$ be a Petri net with a positive marking $\mu_0$. \(\delta\) is the next-state function of $D$. We construct the following context-sensitive grammar $G = (V, X, R, S)$ where $V$ is the set of variables, $X$ is an alphabet, $R$ is a set of productions (rewriting rules) and $S$ is a start symbol, as follows: $V = \{S, [\delta]\} \cup \{[w] | w \in X^2 \cup X^3\} \cup \{[\mu] | \mu \in \Omega_D\} \cup \{[\pi_p] | p \in P\}$ and $R = R_1 \cup R_2 \cup R_3 \cup R_4 \cup R_5 \cup R_6 \cup R_7 \cup R_8$, where

- $R_1 = \{S \rightarrow w | w \in (X \cup X^2 \cup X^3) \cap C(D)\}$,
- $R_2 = \{S \rightarrow [\delta]\mu_0\}$,
- $R_3 = \{[\delta][\nu] \rightarrow [w][\delta][\nu'][\nu'] | \mu \in N^P \cap \Omega_D, w \in X^2 \cup X^3, \nu + \nu' = \delta(\mu, w'), \nu, \nu' \in \Omega_D, \forall w' \in P_r(w), \delta(\mu, w') \in N^P\}$,
- $R_4 = \{[\mu][\nu] \rightarrow [\mu'][\nu'] | \mu + \nu = \mu' + \nu', \mu, \nu, \mu', \nu' \in \Omega_D\}$,
- $R_5 = \{[\delta][\nu] \rightarrow [w][\pi_p] | p \in P, \mu \in N^P \cap \Omega_D, w \in X^2 \cup X^3, \forall w' \in P_r(w) \setminus \{w\}, \delta(\mu, w') \in N^P, \delta(\mu, w)(p) = 0\}$,
- $R_6 = \{[\pi_p][\mu] \rightarrow [\pi_p][\pi_p] | p \in P, \mu \in \Omega_D, \mu(p) = 0\}$,
- $R_7 = \{[w][\pi_p] \rightarrow [\pi_p][w] | p \in P, w \in X^2 \cup X^3\}$,
\[ R_s = \{ [w][\pi_p] \rightarrow w \mid p \in P, w \in X^2 \cup X^3 \} \]

Consequently, \( \mathcal{L}(G) = \mathcal{C}(D) \)
Chapter 5
Maximality of CPN Codes

In Chapter 5 we treat the open problem raised in Chapter 4. A CPN code generated by some input-ordinary Petri net is called an input-ordinary CPN code (iCPNC, for short) and obviously a maximal CPN code. The problem is whether $m\text{CPNC} = i\text{CPNC}$ or not, where $m\text{CPNC}$ (resp., $i\text{CPNC}$) means the family of maximal CPN codes (resp., input-ordinary CPN codes). It is easily seen that the later is a subfamily of the former. But the reverse inclusion is still open in a general Petri net. So we show that the inclusion is true in restricted cases, i.e., the case that the number of places is $\leq 2$, and the case that the number of transitions is equal to 1.

5.1 Fundamental Properties

Here we state some fundamental properties used in the following sections.

\textbf{DEFINITION 5.1.1} Let $PN = (P, X, W, \mu_0)$ be a Petri net and $\mu_0$ be a positive marking. For $w \in X^*$ and $a \in X$, the set $K_w(a)$ of places is defined as follows. If $\delta(\mu_0, w)$ is not defined, then $K_w(a) = \emptyset$. Otherwise,

$K_w(a) = \{p \in P | \delta(\mu_0, w) = \mu, W(a, p) = 0, \exists n \in \mathbb{N}, \ (\mu + n \cdot \Delta(a))(p) = 0, \ \text{and} \ (\mu + n \cdot \Delta(a))(q) \geq 0 \ \text{for} \ \forall q \in P \setminus \{p\}\}.$

An element of $K_w(a)$ is called a critical place (after reading the word $w$). Especially $K_w(a)$ is denoted by $K(a)$ when $w = 1$ (the empty word). $K_w$ is a mapping from $X$ to $2^P$, called the critical place mapping of the Petri net $PN$.

A critical place $p$ of a transition $a$ means that $p$ is a place where the number of tokens first becomes zero when $a$ fires one after another (see Figure 5.1).

\textbf{THEOREM 5.1.1 (Fundamental Theorem)} Let $PN = (P, X, W, \mu_0)$ be a Petri net with a positive marking $\mu_0$, $K$ be its critical place mapping. If $C = \mathbb{C}(PN)$ is a
maximal CPN code, then for any $p \in P$ and $a, b \in X$ the following conditions hold.

1. $p \in K(a)$ implies $W(p, a) \geq W(p, b)$,
2. $p \in K(a) \cap K(b)$ implies $W(p, a) = W(p, b)$.

THEOREM 5.1.2 (Deletion of useless places) Let $PN = (P, X, W, \mu_0)$ be a Petri net with a positive marking \( \mu_0 \), $C = C(PN)$ be a maximal prefix code. Let $p \in P$ be a place such that $\delta(\mu_0, w)(p) \neq 0$ for any $w \in C$. And the Petri net $PN' = (P', X, W', \mu_0')$ is defined as follows:

\[
P' = P \setminus \{p\},
W' \text{ is a restriction of } W \text{ on } P' \\
\mu_0' \text{ is a restriction of } \mu_0 \text{ on } P'.
\]

Then,

$C(PN) = C(PN')$.

$PN'$ is obtained from $PN$ by deleting the place $p$ and its all input/output arcs attached to $p$.

THEOREM 5.1.3 (Reduction rule of two-way arcs) Let $PN = (P, X, W, \mu_0)$ be a Petri net with a positive marking $\mu_0$. Let $C = C(PN)$ be a maximal prefix code. Let $p \in P$, $a \in X$ with $W(p, a) > 0$ and $W(a, p) > 0$. Then the Petri net $PN' = (P, X, W', \mu_0')$
is defined as follows, which is obtained by replacing the weights of the two arcs \((p, a)\) and \((a, p)\).

\[
W(p, a) > W(a, p) \quad \Rightarrow \quad W'(p, a) = W(p, a) - W(a, p), \quad W'(a, p) = 0 \\
W(p, a) = W(a, p) \quad \Rightarrow \quad W'(p, a) = W'(a, p) = 0 \\
W(p, a) < W(a, p) \quad \Rightarrow \quad W'(a, p) = W(a, p) - W(p, a), \quad W'(p, a) = 0 \\
q \neq p \text{ or } b \neq a \quad \Rightarrow \quad W'(b, q) = W(b, q), \quad W'(q, b) = W(q, b)
\]

Then

\[\mathcal{C}(PN) = \mathcal{C}(PN').\]

\[\square\]

**EXAMPLE 5.1.2** Let \(X\) be an alphabet and \(k\) be a positive integer. Suppose that subsets \(X_1\) and \(X_2\) of \(X\) satisfy \(X = X_1 \cup X_2\) and \(X_1 \cap X_2 = \emptyset\). Then, the following language \(C\) is an input-ordinary CPN code.

\[C = \left( \bigcup_{0 \leq i < k} X_2^i X_1 \right) \cup X_2^k.
\]

\[\square\]

Especially, in this example by setting \(X_1 = \emptyset\) and \(X_2 = X\), then \(C = X^k = \{w \in X^* | |w| = k\}\). \(X^k\) is called a (full) uniform code over \(X\). Therefore a uniform code becomes an input-ordinary CPN code.

### 5.1.1 In the case \(|P| = 1\) or \(|X| = 1\)

At first we consider the case the number \(|P|\) of places equals 1 and the case the number \(|X|\) of transitions equals 1.

**THEOREM 5.1.4** Let \(PN = (P, X, W, \mu_0)\) be a Petri net with a positive marking \(\mu_0\). Assume that \(|X| = 1\) or \(|P| = 1\). If \(C = \mathcal{C}(PN)\) is a maximal prefix code, then \(C\) is an input-ordinary CPN code.

Assume that \(|P| = 1\), that is \(P = \{p\}\) in this theorem. Setting \(X_1 = \{a \in X | W(p, a) > 0, W(a, p) = 0\}\) and \(X_2 = X - X_1\), Then

\[C(P, X, W, \mu_0) = (X_1^{n_i-1} \diamond ( \bigcup_{a_i \in X_2} a_i X_i^{n_i}) \circ ) X_1,
\]

where \(n_i = W(a_i, p)/n\), \(\diamond\) is the shuffle product over two languages \(L, K \subset X^*\) defined by \(L \diamond K = \bigcup_{x \in L, y \in K} x \diamond y\), \(x \diamond y = \{x_1 y_1 x_2 y_2 \cdots x_n y_n | x = x_1 x_2 \cdots x_n, y = y_1 y_2 \cdots y_n, \ x_i, y_i \in X^*\ \text{for} \ 1 \leq i \leq n\}\) for \(x, y \in X^*\) and \(L^\circ\) is the shuffle closure of a language \(L\), defined by \(L^\circ = \bigcup_{i \geq 0} L^{\circ i}\), \(L^{\circ 0} = \{1\}\), \(L^{\circ (i+1)} = L^{\circ i} \diamond L\).

In case that a Petri net has only a place or only a transition, we have proved that \(m\text{CPNC}=\text{iCPNC}\). In the following section, we consider the case that a Petri net has two places.
5.2 Maximal CPN Codes with two Places

Here we solve the problem whether \( mCPNC \subseteq iCPNC \) holds or not under the conditions that a Petri net has just two places. All through this section, we assume that a Petri net \( PN = (P, X, W, \mu_0) \) with a positive marking \( \mu_0 \) generating a code satisfies the following conditions without the loss of generality.

1. \( |P| = 2 \), Set \( P = \{p, q\} \).
2. \( X \neq \emptyset \) and \( X \) includes no isolated transition because a marking is unchanged by a isolated transition’s firing.

Moreover if \( C(PN) \) is a maximal CPN code, we implicitly assume that \( PN \) satisfies the next useful conditions.

3. Every arc is one way by THEOREM 5.1.3. That is, for any \( p \in P \) and \( a \in X \),
   \[ W(a, p) = 0 \text{ or } W(p, a) = 0. \]
4. \( PN \) has no useless place by THEOREM 5.1.2. That is, for any \( p \in P \), there exists \( w \in C(PN) \) with \( \delta(\mu_0, w)(p) = 0 \).

5.2.1 Without Source Transitions

In this subsection, each transition in \( X \) is either a sink transition or a transform transition.

**THEOREM 5.2.1** Let \( PN = (P, X, W, \mu_0) \) be a Petri net without source transitions, \( \mu_0 \) be positive and \( P = \{p, q\} \). If \( C = C(PN) \) is a maximal CPN code, then \( C \) is an input-ordinary CPN code.

(proof) Setting \( X_p \) and \( X_q \) as follows:

\[
X_p = \{a \in X \mid p \in K(a)\} = K^{-1}(\{p\}), \\
X_q = \{a \in X \mid q \in K(a)\} = K^{-1}(\{q\}).
\]

(note that \( X_p \cap X_q = K^{-1}(\{p, q\}) = \emptyset \) does not necessarily hold), where \( K \) is the critical place mapping of \( PN \).

Since an arbitrary transition \( a \in X \) is a sink or transform transition by the condition (3), the number of tokens in \( p \) or \( q \) becomes zero when \( a \) fires in succession, that is, \( X = X_p \cup X_q \) holds.

By THEOREM 5s 5.1.1 and 5.1.3 there exist some positive integers \( n_p \) and \( k \) such that \( W(p, a) = n_p \), \( W(a, p) = 0 \) and \( \mu_0(p) = kn_p \) for any \( a \in X_p \). Similarly, there exist some positive integers \( n_q \) and \( l \) such that \( W(q, a) = n_q \), \( W(a, q) = 0 \) and \( \mu_0(q) = ln_q \) for any \( a \in X_q \).

If \( k = l = 1 \), the statement of this theorem holds because the code \( C \) is the uniform code \( X^1 \). If \( X_p = \emptyset \), that is \( X = X_q \), then \( C \) is the uniform code \( C = X^l \in iCPNC \). Similarly \( C \) is also a uniform code \( C = X^k \) if \( X_q = \emptyset \). So we may assume that \( k \cdot l > 1 \), \( X_p \neq \emptyset \) and \( X_q \neq \emptyset \) hold.

If there exists neither \( a \in X_p \) such that \( n_q \nmid W(q, a) \) nor \( b \in X_q \) such that \( n_p \nmid W(p, b) \) or \( a \in X_q \) such that \( n_p \nmid W(p, b) \) or \( n_p \nmid W(a, b) \), then the weight of each output arc is from the place
5.2. Maximal CPN Codes with two Places

p (resp., q) is zero or \( n_p \) (resp., \( n_q \)), the weight of every input arc to \( p \) (resp., \( q \)) is a multiple of \( n_p \) (resp., \( n_q \)). Therefore, \( \mathbb{C}(PN) \) is the same as \( \mathbb{C}(PN') \) generated by the following input-ordinary Petri net \( PN' = (P', X', W', \mu'_0) \):

\[
P' = P = \{p, q\}, \quad X' = X,
\]

\[
W'(p, a) = W(p, a)/n_p \in \{0, 1\}, \quad W'(a, p) = W(a, p)/n_p \in N_0 \quad \text{for} \quad \forall a \in X,
\]

\[
W'(q, a) = W(q, a)/n_q \in \{0, 1\}, \quad W'(a, q) = W(a, q)/n_q \in N_0 \quad \text{for} \quad \forall a \in X,
\]

\[
\mu'_0(p) = \mu_0(p)/n_p = k, \quad \mu'_0(q) = \mu_0(q)/n_q = l.
\]

Hence, \( \mathbb{C}(PN) \) is an input-ordinary CPN code.

The remaining cases are that there exists \( a \in X_p \) such that \( n_q \frac{1}{n} W(q, a) \) or \( n_q \frac{1}{n} W(a, q) \), and that there exists \( b \in X_q \) such that \( n_p \frac{1}{n} W(p, b) \) or \( n_p \frac{1}{n} W(b, p) \). By considering the symmetry, we must check the next the next cases:

(A) \( \exists a \in X_p, \exists b \in X_q \{x = W(p, b) > 0, y = W(q, a) > 0, \text{ and } (n_p \perp x or n_q \perp y)\}\)

(B) \( \exists a \in X_p, \exists b \in X_q \{x = W(b, p) > 0, y = W(q, a) > 0, \text{ and } (n_p \perp x or n_q \perp y)\}\)

(C) \( \exists a \in X_p, \exists b \in X_q \{W(b, p) = 0, y = W(q, a) > 0, \text{ and } n_q \perp y\}\)

(D) \( \exists a \in X_p, \exists b \in X_q \{x = W(b, p) > 0, y = W(a, q) > 0, \text{ and } (n_p \perp x or n_q \perp y)\}\)

(E) \( \exists a \in X_p, \exists b \in X_q \{x = W(b, p) > 0, W(q, a) = 0, \text{ and } n_p \perp x\}\)

By LEMMA 5.2.1, 5.2.2, 5.2.3 and 5.2.5, we can show that \( C \) is an input-ordinary CPN code in case of (A), (B), (C) or (E), respectively. On the other hand the case (D) does not happen because \( C \) is not a maximal CPN code by LEMMA 5.2.4.

We state the LEMMA 5.2.1~5.2.5 in referred in the proof of THEOREM 5.2.1.

**LEMMA 5.2.1** Let \( PN = (P, X, W, \mu_0) \) be a Petri net with a positive marking \( \mu_0 \) which is satisfied the condition (A) in the proof of THEOREM 5.2.1. If \( C = \mathbb{C}(PN) \neq \emptyset \) is a maximal CPN code, then it is a uniform code \( X^k \), that is, an input-ordinary CPN code.

**LEMMA 5.2.2** Let \( PN = (P, X, W, \mu_0) \) be a Petri net with a positive marking \( \mu_0 \) which is satisfied the condition (B) in the proof of THEOREM 5.2.1. If \( C = \mathbb{C}(PN) \neq \emptyset \) is a maximal CPN code, then it is an input-ordinary CPN code.

**LEMMA 5.2.3** Let \( PN = (P, X, W, \mu_0) \) be a Petri net with a positive marking \( \mu_0 \) which is satisfied the condition (C) in the proof of THEOREM 5.2.1. If \( C = \mathbb{C}(PN) \neq \emptyset \) is a maximal CPN code, then it is an input-ordinary CPN code.

**LEMMA 5.2.4** Let \( PN = (P, X, W, \mu_0) \) be a Petri net with a positive marking \( \mu_0 \) which is satisfied the condition (D) in the proof of THEOREM 5.2.1. If \( C = \mathbb{C}(PN) \neq \emptyset \) cannot be a maximal CPN code.

**LEMMA 5.2.5** Let \( PN = (P, X, W, \mu_0) \) be a Petri net with a positive marking \( \mu_0 \) which is satisfied the condition (E) in the proof of THEOREM 5.2.1. If \( C = \mathbb{C}(PN) \neq \emptyset \) is a maximal CPN code, then it is an input-ordinary CPN code.
5.2. With at least one Source Transitions

In this subsection we show that the code which is generated by a Petri net with two places and at least one source transitions.

**REMARK 5.2.1** A Petri net \( PN = (P, X, W, \mu_0) \) is called semi-input-ordinary if the following condition is satisfied.

For each place \( p \), there exists a positive integer \( n_p \) such that \( W(p, a) = 0 \) or \( = n_p \) and \( W(a, p) \) is a multiple of \( n_p \) for any transition \( a \in X \), and \( \mu_0(p) \) is a multiple of \( n_p \).

If a Petri net \( PN = (P, X, W, \mu_0) \) is semi-input-ordinary, then the code \( C(PN) \) is obviously an input-ordinary CPN code.

**DEFINITION 5.2.1** Let \( PN = (P, X, W, \mu_0) \) be a Petri net. A place \( p \in P \) is controllable if there are a source transition \( c \in X \) and a sink or transform transition \( a \in X \) satisfying either of the following two conditions (a) or (b) for any place \( q \in P \setminus \{p\} \).

(a) \( x > 0 \) and \( xv - uy > 0 \),

(b) \( x > 0, u > 0, y \neq 0 \) and \( v = 0 \),

where \( x = W(p, a), y = W(q, a), u = W(c, p), v = W(c, q) \) and \( y^* = W(a, q) \). Otherwise \( p \) is called uncontrollable.
For example, in case of $|P| = 2$, there are the fifteen ways to give weights on the arcs among arbitrary two places $p$ and $q$, a source transition $c \in X$ and a sink or transform transition $a \in X$.

**FACT** Let $u$ and $x$ be nonnegative integers and $d = \gcd(u, x)$ be the greatest common divisor of $u$ and $x$ (note that $\gcd(0, x) = x$ and $\gcd(u, 0) = u$). Then $us + xt = d$ for some integers $s$ and $t$. \hfill \Box

**LEMMA 5.2.6** Let $PN = (P, X, W, \mu_0)$ be a Petri net with source transitions. If a place $p \in P$ be controllable, then the following conditions hold:

For arbitrary nonnegative integers $i$ and $j$ with $\mu_0(p) - d \times i \geq 0$, there is a word $w \in X^+$ such that

\[
\delta(\mu_0, w)(p) = \mu_0(p) - d \times i, \\
\delta(\mu_0, w)(q) \geq \mu_0(q) + j, \quad \text{for } \forall q \in P \setminus \{p\}.
\]

\hfill \Box

The next theorem holds regardless of the number $|P|$ of places.

**LEMMA 5.2.7** Let $PN = (P, X, W, \mu_0)$ be a Petri net with source transitions and $\mu_0$ be a positive marking. Let $C = \mathbb{C}(PN)$ be a maximal CPN code. If a place $p \in P$ is controllable, the following conditions hold:

1. There exists some positive integer $n_p$ such that $W(p, a) = 0$ or $W(p, a) = n_p$ for any $a \in X$.
2. $n_p | W(a, p)$ for any $a \in X$.
3. $n_p | \mu_0(p)$.

\hfill \Box

**COROLLARY 5.2.1** Let $PN = (P, X, W, \mu_0)$ be a Petri net with at least one source transitions and $\mu_0$ be a positive marking. If each $p \in P$ is controllable and $C = \mathbb{C}(PN)$ is a maximal CPN code, then $C$ is an input-ordinary CPN code. \hfill \Box

Note that the COROLLARY 5.2.1 is true independently to the number of places. Then we check the case that both two places are uncontrollable and the case that one place is controllable but the other is not. The remaining cases needs the restriction that the number of places is two.

**LEMMA 5.2.8** Let $PN = (P, X, W, \mu_0)$ be a Petri net with at least one source transitions, $\mu_0$ be a positive marking and $|P| = 2$ ($P = \{p, q\}$). If each place is uncontrollable and $C = \mathbb{C}(PN)$ is a maximal CPN code, then $C$ is an input-ordinary CPN code. \hfill \Box

**LEMMA 5.2.9** Let $PN = (P, X, W, \mu_0)$ be a Petri net with source transitions, $\mu_0$ be a positive marking and $|P| = 2$ ($P = \{p, q\}$). One place $p$ is controllable and the other place $q$ is not. If $C = \mathbb{C}(PN)$ be a maximal CPN code, then $C$ is an input-ordinary CPN code. \hfill \Box
The LEMMAs 5.2.7 to 5.2.9 that are stated above derives the next theorem 5.2.2

**THEOREM 5.2.2** Let $PN = (P, X, W, \mu_0)$ be a Petri net with source transitions, $\mu_0$ be positive and $|P| = 2$. If $C = C(PN)$ is a maximal CPN code, then $C$ is an input-ordinary CPN code. \(\Box\)

We obtain the final result of this chapter from the THEOREMs 5.2.1 and 5.2.2.

**THEOREM 5.2.3** Let $PN = (P, X, W, \mu_0)$ be a Petri net, $\mu_0$ be positive and $|P| = 2$. If $C = C(PN)$ is a maximal CPN code, then $C$ is an input-ordinary CPN code. \(\Box\)
Chapter 6

Conclusion

Recently Petri nets are used not only as technical modeling tools for parallel/concurrent systems, but also as theoretical model of computation like automata, language generators, grammar controllers, and so on. Petri net theory is one of advanced and interested fields in automata, formal languages and computation. In this literature we treated two topics, the Petri net structures (in Chapter 3) and Petri net codes (in Chapters 4 and 5).

In Chapter 3, the notion of automorphism group of a Petri net structure was newly introduced. We showed the main theorem that for a given finite group $G$, there exists a Petri net structure $N$, called a transformation net, such that $\text{Aut}(N)$ is isomorphic to $G$. The structure $N$ corresponds to the right regular representation of $G$.

The four (S-, D-, C-, B-) types of Petri net codes, which are all prefix codes, were introduced as similar way to define Petri net languages. Mainly we use Petri nets as accepters of codes and treat firing sequences themselves without labeling functions. In Chapters 4 and 5, C-type Petri net (CPN, for short) codes are mainly focused. The CPN code $C(N, \mu_0)$ generated by a Petri net $(N, \mu_0)$ is the set of all nonpositive firing sequences in $(N, \mu_0)$ whose proper prefixes are all positive firing sequences instead. That is, $C(N, \mu_0) = L \setminus LX^+$, where $L = L(N, \mu_0) \setminus L_+(N, \mu_0)$. If a CPN code is a maximal prefix code, then we call it a maximal CPN code.

In the first half of Chapter 4, various properties of finite maximal CPN codes were investigated and two operations $\oplus$(some kind of parallel operation) and $\otimes$(some kind of interruption) were introduced.

The property being a maximal CPN code over $X$ is not preserved under concatenation, $\oplus$ and $\lambda$-free homomorphism but is preserved under $\otimes$. In the second half, we investigated the generative power of CPN codes. There it is shown that there exists a CPN code which is not context-free, but arbitrary CPN code is a context-sensitive language.

In Chapter 5 we considered the open problem raised in Chapter 4. That is, whether the family $\text{mCPNC}$ stated above is included in the family $\text{iCPNC}$ of CPN codes which are generated by some input-ordinary Petri nets.

The notion of maximality of a CPN code is very important in relation to liveness or deadlock. $C(N, \mu_0)$ is a maximal prefix code or $C(N, \mu_0) = \emptyset$ if and only if all of transitions
are enabled under a marking reachable from $\mu_0$ through a positive firing sequence in $(N, \mu_0)$. This condition is obviously true if $(N, \mu_0)$ is input-ordinary. Conversely we wonder whether the set of all positive firing sequences in a general Petri net $(N, \mu_0)$ with $\mathbb{C}(N, \mu_0)$ being a maximal prefix code is identical with the set of all positive firing sequences in some input-ordinary Petri net $(N_1, \mu_1)$, that is, $L_+(N, \mu_0) = L_+(N_1, \mu_1)$.

We proved that $\text{mCPNC} = \text{iCPNC}$ is true in restricted cases, i.e., in the case that the number of places is $\leq 2$, and in the case that the number of transitions is equal to 1. It still remains open in a general Petri net.
Bibliography


Part III

Publications
1. Publications related to the dissertation

Chapter 4

a residue class group," TECHNICAL REPORT OF IEICE, COMP95-71:47--54,


Chapter 5

[4] M. Ito and Y. Kunimochi, "CPN Languages and Codes,

[5] M. Ito and Y. Kunimochi, "Some Petri Net Languages and Codes,

Chapter 6

Technical Report kokyuroku 1503, RIMS, Kyoto University, pp. 139-147, 2006.7.
京都大学数理解研究所講究録1503, pp. 139-147, 2006.7.

[7] Y. Kunimochi, "Place Dependency of a Petri Net Generating a Maximal Prefix Code,

2. Other publications

[8] Y. Kunimochi,
"Some Remarks on Hypercodes Defined on Petri Nets," The Bulletin of the


[17] T. Tamama, Y. Kunimochi and H. Hakamata, 
"Design and Test Fabrication of a 32-bit RISC processor based on a Dataflow Machine Architecture," 
The Bulletin of the Shizuoka Institute of Sci. and Tech., 
玉真昭男，國持良行，袴田祐幸，
"データフローエキテクチャに基づいた 32 ピット RISC プロセッサの設計と試作，" 

[18] 國持良行，"ベトリネットが生成する極大プレフィクスコードの性質について，" 
Y. Kunimochi,"Some Properties of Maximal Prefix Codes Generated by Petri Nets," 
The Bulletin of the Shizuoka Institute of Sci. and Tech., 

[19] 國持 良行, 関本 彰次, 
"データフローエキテクチャを用いた並列リダクションマシンの実現について，" 

[20] 幸谷智紀，國持良行，菅沼義昇，"静岡理工科大学における遠隔講義実験について" 