Finite Automata Based on Quantum Logic and Their Determinization *

Yongming Li
College of Computer Science, Shaanxi Normal University, Xi'an, 710062, China
Email: liyongm@snnu.edu.cn, Tel.: +862985310166, Fax: +862985310161

Abstract
We give the quantum subset construction of orthomodular lattice-valued finite automata, then we show the equivalence between orthomodular lattice-valued finite automata, orthomodular lattice-valued deterministic finite automata and orthomodular lattice-valued finite automata with empty string-moves. Based on these equivalences, we study the algebraic operations on orthomodular lattice-valued regular languages, then we establish Kleene theorem in the frame of quantum logic.

Keywords: Quantum logic, finite automata; subset construction; quantum language; determinization; Kleene Theorem.

1 Introduction
In classical computation theory, characterizing all formal languages, or even better sorting them in some hierarchy, was an important issue. For example, the most restricted class-the regular languages-can be characterized by finite automata and by regular expressions. It is well-known that regular languages can be recognized by deterministic finite automata, nondeterministic finite automata (with or without empty string-moves), the technique to prove the equivalence between nondeterministic and deterministic automata is the subset construction [10, 5, 7]. Another important result in classical automata theory is the Kleene theorem which shows the equivalence between finite automata and regular expressions. All these results have been extended to fuzzy finite automata as the fuzzy computing models by introducing fuzzy subset construction and fuzzy regular expressions, see [10] for the detail. In the frame of weighted automata theory, the subset construction and Kleene theorem have been also discussed, see

---

*This work is supported by National Science Foundation of China (Grant No.10571112), National 973 Foundation Research Program (Grant No.2002CB312200) and Key Research Project of Ministry of Education of China (No.107106).
for the detail. The subset construction and Kleene theorem form the basic results for the computational models with different purposes. From this core the theory has developed into numerous directions. Computing models of quantum computation is a new research along this direction.

The ideas of quantum computing came from the connections between physics and computation [12]. In particular, in 1994 Shor discovered a polynomial-time algorithm for factoring prime on quantum computers, and Grover then found an algorithm for searching through a database in square root time. Since then, quantum computing has attracted more and more attention in the research community. In this field, the computing models of quantum computation is still one of the most important topic to study. Quantum finite automata can be viewed as a kind of quantum computer model with finite memory, for which we may refer to ref. [4, 11, 1]. A more fundamental issue regarding quantum computing models may be automata theory based on quantum logic [17, 18, 19, 13, 14, 15] (called orthomodular lattice-valued automata). Quantum logic was suggested by Birkhoff and Neumann in 1936 for studying the logical basic of quantum mechanics, and it originated from the Hilbert space’s formalization of quantum mechanics. Since a state of a quantum system can be described by a closed subspace of a Hilbert space, while all closed subspaces of a Hilbert space are endowed with the algebraic structure of orthomodular lattices, it was proposed that orthomodular lattices were thought of as the algebraic version of quantum logic. Actually, orthomodular lattices sometimes are defined directly as quantum logic. Thus, investigating orthomodular lattice-valued automata may be considered to be an important aspect of the logical basic of quantum computing. Recently, the author [17, 18, 19] primarily and very significantly considered automata theory based on quantum logic (l-valued automata), in which quantum logic is understood as a logic whose truth-value set is an orthomodular lattice, and an element of an orthomodular lattice is assigned to each transition of an automaton and it is considered to be the truth value of the proposition describing the transition. This is a logical approach to quantum computation in which the ultimately objective is to manage to set up the logic platform for the quantum computation, and it should be treated as a further abstraction of mathematical models of quantum computation. With this approach, the author dealt with some operations on l-valued automata, and interestingly established corresponding pumping lemma, showed the equivalence between the distributivity of truth-value lattices and the product operation of orthomodular lattice-valued automata, etc., showed an essential difference exists between the classical theory of computation and the computation theory based on quantum logic.

The concept of an orthomodular lattice-valued finite automaton is a natural generalization of
the concept of a nondeterministic automaton, as the concepts of an orthomodular lattice-valued set and an orthomodular lattice-valued relation are generalizations of the classical concepts of a set and a relation. Relationships between orthomodular lattice-valued nondeterministic and deterministic automata have been studied by Ying [19]. The method for determinization of orthomodular lattice-valued automata used by Ying in [19] is analogous to the well-known subset construction, and it is called here the extended subset construction. Unfortunately, extended subset construction does not work well for orthomodular lattice-valued finite automaton. As shown by Ying in [19], under extended subset construction, one can not prove the equivalence between orthomodular lattice-valued nondeterministic and deterministic automata. In fact, Ying proved that the equivalence between orthomodular lattice-valued nondeterministic and deterministic automata under extended subset construction is equivalent to the underlying logic being classical logic (i.e., the used truth structure as an orthomodular lattice must be a Boolean algebra). It is left open as a problem whether orthomodular lattice-valued nondeterministic and deterministic automata are equivalent. We shall introduce quantum subset construction in this paper to study this problem. Indeed, using the quantum subset construction introduced in this paper, we show the equivalence between orthomodular lattice-valued nondeterministic (with or without empty string-moves) and deterministic automata. Furthermore, we characterize the quantum languages recognized by orthomodular lattice-valued automata by the orthomodular lattice-valued recognizable step languages in Theorem 3.1 which have very simple construction. Using this characterization of recognizable quantum languages, we further show the Kleene theorem holds in the frame of quantum logic. Many results in [19] can be strengthen in this manner.

The content of this paper is arranged as follows. In Section 2, we first recall the definition of orthomodular lattice-valued automata, then we introduce the notion of orthomodular lattice-valued deterministic automata. By introducing the quantum subset construction, we prove the equivalence between orthomodular lattice-valued nondeterministic (with or without empty string-moves) automata and deterministic automata. In Section 3, we first give a simple characterization of quantum regular languages, the operations property of quantum regular languages is discussed. Then the Kleene theorem in quantum logic is presented. Some conclusion is presented finally.
2 Determinization of \(l\)-valued finite automata and quantum subset constructions

Quantum logic is understood as a (complete) orthomodular lattice-valued logic, for the detail, we refer to \[6, 15, 19\]. We briefly recall some notions and notations of quantum logic. An ortholattice is a 7-tuple \(l = (L, \leq, \land, \lor, \bot, 0, 1)\), where \(l = (L, \leq, \land, \lor, \bot, 0, 1)\) is a bounded lattice, 0 and 1 are the least and largest elements of \(L\), respectively, \(\leq\) is the partial ordering in \(L\); and for any \(a, b \in L\), \(a \land b\) and \(a \lor b\) stand for the greatest lower bound (or meet) and the least upper bound (or join) of \(a\) and \(b\), respectively. \(\bot\) is a unary operation on \(L\), called orthocomplement, and required to satisfy the following conditions: for any \(a, b \in L\), \(a \land a^\perp = 0\), \(a \lor a^\perp = 1\); \(a^\perp \perp = a\); \(a \leq b\) implies \(b^\perp \leq a^\perp\). An orthomodular lattice is an ortholattice \(l = (L, \leq, \land, \lor, \bot, 0, 1)\) satisfying the orthomodular law: for all \(a, b \in L\), \(a \leq b\) implies \(a \land (a^\perp \lor b) = b\). A quantum logic is a (complete) orthomodular lattice-valued logic (called \(l\)-valued logic). Defined an implication operator \(\rightarrow\) on \(l\) satisfying: for all \(a, b \in L\), \(a \leq b\) if and only if (iff) \(a \rightarrow b = 1\). In this paper, we use Sasaki arrow as the implication operator. Sasaki arrow is defined as follows: for all \(a, b \in L\), \(a \rightarrow b = a^\perp \lor (a \land b)\). The bi-implication operator \(\leftrightarrow\) is defined as follows: for all \(a, b \in L\), \(a \leftrightarrow b = (a \rightarrow b) \land (b \rightarrow a)\). The syntax of \(l\)-valued logic is similar to that of classical first-order logic. We have three primitive connectives \(\neg\) (negation), \(\lor\) (conjunction) and \(\rightarrow\) (implication), and a primitive quantifier \(\exists\) (existential quantifier). The connectives \(\land\) (conjunction) and \(\leftrightarrow\) (bi-implication) and the universal quantifier \(\forall\) are defined in terms of \(\neg\), \(\lor\), \(\rightarrow\) and \(\exists\) in the usual way. In addition, we need to use some set-theoretical formulas. Let \(\in\) (membership) be a binary (primitive) predicate symbol. Then \(\subseteq\) and \(\equiv\) (equality) can be defined with \(\in\) as usual. The semantics of \(l\)-valued logic is given by interpreting the connectives \(\neg\), \(\lor\) and \(\rightarrow\) as the operations \(\bot\), \(\lor\) and \(\rightarrow\) on \(L\), respectively, and interpreting the quantifier \(\exists\) as the least upper bound in \(l\). Moreover, the truth value of set-theoretical formula \(x \in A\) is \([x \in A] = A(x)\). In the \(l\)-valued logic, 1 is the unique designated truth value; a formula \(\varphi\) is valid iff \([\varphi] = 1\), and denoted by \(\models_l \varphi\). For a finite subset \(X\) of \(l\), the (commutator) \(\gamma(X)\) generated by \(X\) is defined as follows:
\[
\gamma(X) = \lor\{\land_{a \in X} a^{f(a)} : f : X \rightarrow \{1, -1\}\} \text{ is a mapping},
\]
where, \(x^1 = x\), \(x^{-1} = x^\perp\).

In order to distinguish the symbols representing languages and the symbols representing lattices, we use symbol \(l\) to represent orthomodular lattice, and use \(L\) to represent language. We use the symbols \(a, b, c, d, k\) to represent the elements of \(l\).

**Definition 2.1.** \[19\] An \(l\)-valued finite automaton (\(l\)-VFA for short) is a 5-tuple \(\mathcal{A} = (Q, \Sigma, \delta, I, F)\),
where $Q$ denotes a finite set of states, $\Sigma$ a finite input alphabet, and $\delta$ is an $l$-valued subset of $Q \times \Sigma \times Q$; that is, a mapping from $Q \times \Sigma \times Q$ into $l$, and it is called the $l$-valued (quantum) transition relation. Intuitively, $\delta$ is an $l$-valued (ternary) predicate over $Q$, $\Sigma$ and $Q$, and for any $p, q \in Q$ and $\sigma \in \Sigma$, $\delta(p, \sigma, q)$ stands for the truth value (in quantum logic) of the proposition that input $\sigma$ causes state $p$ to become $q$. $I$ and $F$ are $l$-valued subset of $Q$, which represent the initial state and final states, respectively. For each $q \in Q$, $I(q)$ indicates the truth value (in the underlying quantum logic) of the proposition that $q$ is an initial state, $F(q)$ expresses the truth value (in quantum logic) of the proposition that $q$ is a final state.

The propositions of the form

“$q$ is an initial state”, written “$q \in I$”.

“$q$ is a final state”, written “$q \in F$”.

“input $\sigma$ causes state $q$ to become $p$, according to the specification given by $\delta$”, written “$(q, \sigma, p) \in \delta$”.

denote the atomic propositions in our logical languages designated for describing $l$-valued automaton $A$. The truth values of the above three propositions are respectively $I(q)$, $F(q)$ and $\delta(q, \sigma, p)$. We use the symbols $\sigma, \tau$ to represent the elements in $\Sigma$, use the symbols $\omega, \theta$ to denote the strings over $\Sigma$, and use $\varepsilon$ to represent the empty string over $\Sigma$. We use the symbols $A, B$ to denote the $l$-valued finite automata.

For an $l$-VFA $A$, the $l$-valued unary recognizability predicate $rec_A$ over $\Sigma^*$ is defined as a mapping from $\Sigma^*$ into $l$: for each $\omega \in \Sigma^*$, let $\omega = \sigma_1 \cdots \sigma_n$ for some $n \geq 0$,

\[ rec_A(\omega) = (\exists q_0 \in Q) \cdots (\exists q_n \in Q). (q_0 \in I \land q_n \in F \land (q_0, \sigma_1, q_1) \in \delta \land \cdots \land (q_n-1, \sigma_n, q_n) \in \delta). \]

In other words, the truth value of the proposition that $\omega$ is recognizable by $A$ is given by

\[ [rec_A(\omega)] = \bigvee \{ I(q_0) \land \delta(q_0, \sigma_1, q_1) \land \cdots \land \delta(q_{n-1}, \sigma_n, q_n) \land F(q_n) : q_0, \cdots, q_n \in Q \}. \]

We call $rec_A$ the $l$-valued language recognized or accepted by $l$-VFA $A$. We use $l(\Sigma^*)$ to denote the set of all $l$-valued language over $\Sigma^*$, which is an $l$-valued subset of $\Sigma^*$; that is, a mapping from $\Sigma^*$ to $l$. We also call $l$-valued languages by quantum languages. For an $A \in l(\Sigma^*)$, if there is an $l$-VFA $A$ such that $A = rec_A$, then we call $A$ an $l$-valued regular language or $l$-regular language on $\Sigma$, which is also called quantum regular language without mentioned the truth-valued lattice.

Furthermore, we can define unary predicate $Rec_{\Sigma}$ on $l(\Sigma^*)$ as follows: for all $B \in l(\Sigma^*)$,

\[ Rec_{\Sigma}(B) = (\exists A \in A (\Sigma)). (B \equiv rec_A). \]

where $A (\Sigma)$ writes for the class of all $l$-valued automata over $\Sigma$, we refer to [19] for the detail.
First, we show that the image set of each quantum regular language is always a finite set of \( l \).

**Lemma 2.1.** \([9]\) Let \( l \) be a lattice, and \( X \) a finite subset of \( l \). Then the \( \wedge \)-semilattice of \( l \) generated by \( X \), written as \( X_\wedge \), is finite, the \( \vee \)-semilattice of \( l \) generated by \( X \), denoted \( X_\vee \), is also finite, where \( X_\wedge = \{ x_1 \land \cdots \land x_k : k \geq 1, x_1, \ldots, x_k \in X \} \cup \{ 1 \} \), and \( X_\vee = \{ x_1 \lor \cdots \lor x_k : k \geq 1, x_1, \ldots, x_k \in X \} \cup \{ 0 \} \).

**Proposition 2.1.** Let \( \mathcal{A} = (Q, \Sigma, \delta, I, F) \) be an \( l \)-VFA. Then the image set of the quantum language \( \text{rec}_\mathcal{A} \), as a mapping from \( \Sigma^* \) to \( l \), is finite; that is, the subset \( \text{Im}(\text{rec}_\mathcal{A}) = \{ r \in l : \exists \omega \in \Sigma^*, [\text{rec}_\mathcal{A}(\omega)] = r \} \) of \( l \) is finite.

**Proof** For any \( \omega = \sigma_1 \cdots \sigma_k \in \Sigma^* \), observing that \( [\text{rec}_\mathcal{A}(\omega)] = \bigvee \{ I(q_0) \land \delta(q_0, \sigma_1, q_1) \land \cdots \land \delta(q_{k-1}, \sigma_k, q_k) \land F(q_k) : q_0, \ldots, q_k \in Q \} \). On input \( \omega = \sigma_1 \cdots \sigma_k \in \Sigma^* \), there are only finite accepting paths, assumed as \( m \), causing an initial state \( q_0 \in I \) to become a final state \( q_k \in F \). For the \( i \)-th accepting path, we let \( a_{i0} = I(q_0), a_{i1} = \delta(q_0, \sigma_1, q_1), \ldots, a_{ik} = \delta(q_{k-1}, \sigma_k, q_k) \) and \( a_{i,k+1} = F(q_k) \). Then the truth value of \( \text{rec}_\mathcal{A}(\omega) \) can be calculated as, \([\text{rec}_\mathcal{A}(\omega)] = (a_{10} \land \cdots \land a_{1k} \lor a_{1,k+1}) \lor \cdots \lor (a_{m0} \land \cdots \land a_{mk} \land a_{m,k+1})\). Let \( X = \text{Im}(\delta) \cup \text{Im}(I) \cup \text{Im}(F) \), then \( X \) is obvious a finite subset of \( l \) and \( a_{ij} \in X \) for any \( 1 \leq i \leq m \) and \( 0 \leq j \leq k + 1 \). For any \( \omega \in \Sigma^* \), by the above observation, we know that \([\text{rec}_\mathcal{A}(\omega)] \in (X_\wedge)_\vee \), so \( \text{Im}(\text{rec}_\mathcal{A}) \subseteq (X_\wedge)_\vee \). By Lemma 2.2 \((X_\wedge)_\vee \) is a finite subset of \( l \), and thus \( \text{Im}(\text{rec}_\mathcal{A}) \), as a subset of \((X_\wedge)_\vee \), is also a finite subset of \( l \). 

Due to Proposition 2.1, for any \( l \)-VFA, the image set of its recognizable quantum language is always finite. Then we have the following observation: the orthomodular lattice \( l \) may be infinite as a set, but for a given \( l \)-VFA \( \mathcal{A} \), only a finite subset of \( l \) is employed in the operating of \( \mathcal{A} \). This observation is the core in the introducing of quantum subset construction in this section.

The notion of nondeterminism plays a central role in the theory of computation. The nondeterministic mechanism enables a device to change its states in a way that is only partially determined by the current state and the input symbol. The concept of \( l \)-VFA is obviously a generalization of nondeterministic finite automaton (NFA for short). In classical theory of automata, each nondeterministic finite automaton is equivalent to a deterministic one; more precisely, there exists a deterministic finite automaton (DFA for short) which accepts the same language as the originally given nondeterministic one does. The construction of DFA from an NFA is the well-know subset construction introduced by Rabin and Scott \([16]\). With respect to the case of \( l \)-VFA, the situation is more complex. In fact, as shown in \([19]\), the subset
construction does not work well for l-VFA. That is, for an l-VFA $A$, one can construct an $l$-valued deterministic finite automaton $B$, as defined in [19] using the subset construction. However, $B$ is not necessarily equivalent to $A$, i.e., the equality $\text{rec}_A = \text{rec}_B$ does not hold in general. Some conditions that guarantee the equivalence between $A$ and $B$ are given in [19]. Therefore, it is an open problem whether an l-VFA can always be determinizable. We shall show that the answer is affirmative. We shall introduce subset construction in the frame of quantum logic which we call it the quantum subset construction. First, we define a new kind of deterministic l-VFA, which is stronger than that given in [19] using the same name. We require some stronger condition for the quantum transition.

**Definition 2.2.** An l-valued deterministic finite automaton (l-V DFA for short) is a 5-tuple $A = (Q, \Sigma, \delta, q_0, F)$, where $Q$, $\Sigma$ and $F$ are the same as in an l-valued automaton, $q_0 \in Q$ is the initial state, and the quantum transition relation $\delta$ is crisp and deterministic; that is, $\delta$ is a mapping from $Q \times \Sigma$ into $Q$.

Note that our definition differs from the usual definition of a deterministic automaton only in that the final states form an l-valued subset of $Q$. This, however, makes it possible to accept words to certain truth degrees (in the underlying quantum logic), and thus to recognize quantum languages.

For an l-V DFA, $A = (Q, \Sigma, \delta, q_0, F)$, its corresponding l-valued recognizability predicate $\text{rec}_A \in l(\Sigma^*)$ is defined as: for all $\omega = \sigma_1 \cdots \sigma_n \in \Sigma^*$,

$$\text{rec}_A(\omega) = (\exists q_1 \in Q) \cdots (\exists q_n \in Q). (q_0 \in F \land \delta(q_0, \sigma_1) = q_1 \land \cdots \land \delta(q_{n-1}, \sigma_n) = q_n).$$

Write $\delta^*$ the extension of transition relation $\delta$ by putting $\delta^*(q, \varepsilon) = q$ and $\delta^*(q, \omega\sigma) = \delta(\delta^*(q, \omega), \sigma)$ for any $q \in Q$ and $\omega \in \Sigma^*$ and $\sigma \in \Sigma$, then the truth value of the proposition $\text{rec}_A(\omega)$ is given by,

$$[\text{rec}_A(\omega)] = F(\delta^*(q_0, \omega)).$$

Obviously, the notion of l-V DFA is a special case of l-valued deterministic automata defined in [19], but the converse inclusion does not hold in general.

For any l-VFA, $A = (Q, \Sigma, \delta, q_0, F)$, we now introduce the quantum subset construction to construct an equivalent l-V DFA $A^d = (Q^d, \Sigma, \eta, S, E)$ from $A$.

Let $X = \text{Im}(\delta) \cup \text{Im}(I) \cup \text{Im}(F)$, then $X$ is obvious a finite subset of $l$. Let $l_1 = X_\Lambda$. By Lemma 2.1, $l_1$ is a $\land$-semilattice of $l$ generated by $X$ and is also finite subset of $l$. Choose $Q^d = 2^{Q \times (l_1 - \{0\})}$, where $2^{Q \times (l_1 - \{0\})}$ denotes the set of all subsets of $Q \times (l_1 - \{0\})$. Then $Q^d$ is obvious a finite set. Take
then $S \subseteq Q^d$. The state transition relation $\eta : Q^d \times \Sigma \rightarrow Q^d$ is defined as, for any $(q, r) \in Q \times (l_1 - \{0\})$ and $\sigma \in \Sigma$,

$$\eta((q, r), \sigma) = \{(p, \delta(q, \sigma, p) \land r) : p \in Q \land \delta(q, \sigma, p) \land r \neq 0\},$$

and for $Z \subseteq Q^d$,

$$\eta(Z, \sigma) = \bigcup \{\eta((q, r), \sigma) : (q, r) \in Z\}.$$

By the definition of $l_1$, $l_1$ is closed under finite meet operation, i.e., for any $a, b \in l_1$, $a \land b \in l_1$, it follows that, for any $r \in l_1$ and for any $(p, \sigma, q) \in Q \times \Sigma \times Q$, $r \land \delta(p, \sigma, q) \in l_1$, and thus $\eta((q, r), \sigma) \in Q^d$ for any $(q, r) \in Q \times (l_1 - \{0\})$. Then the mapping $\eta$ is well defined. The $l$-valued final state $E : Q^d \rightarrow l$ is defined by, for any $Z \subseteq Q^d$,

$$E(Z) = \bigvee \{r \land F(q) : (q, r) \in Z\}.$$

Then $\mathcal{A}^d$ is an $l$-VFA.

**Theorem 2.1.** For any $l$-VFA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$, the $l$-VFA $\mathcal{A}^d = (Q^d, \Sigma, \eta, S, E)$ constructed above is equivalent to $\mathcal{A}$, i.e., $\text{rec}_\mathcal{A} = \text{rec}_\mathcal{A}^d$. In the language of quantum logic, it means that, for any $\omega \in \Sigma^*$,

$$\models_I \text{rec}_\mathcal{A}(\omega) \leftrightarrow \text{rec}_\mathcal{A}^d(\omega).$$

**Proof.** We wish to show by induction on the length $|\omega|$ of input string $\omega$ that $\eta^*(S, \omega) = \{\eta^*(S, \omega) = \{(q_n, I(q_0) \land \delta(q_0, \sigma_1, q_1) \land \cdots \land \delta(q_{n-1}, \sigma_n, q_n)) : q_0, \ldots, q_n \in Q$ and $r_n = I(q_0) \land \delta(q_0, \sigma_1, q_1) \land \cdots \land \delta(q_{n-1}, \sigma_n, q_n) \neq 0\},$ where $\omega = \sigma_1 \cdots \sigma_n$. For $n = 0$, the result is trivial since $\omega = \varepsilon$ and $\eta^*(S, \varepsilon) = \{(q_0, I(q_0)) : q_0 \in Q$ and $I(q_0) \neq 0\}$. Suppose that the hypothesis is true for strings of length $n$ or less. Let $\omega = \sigma_1 \cdots \sigma_{n+1}$ be a string of length $n + 1$, write $x = \sigma_1 \cdots \sigma_n$, then $\omega = x \sigma_{n+1}$. Then

$$\eta^*(S, x \sigma_{n+1}) = \eta^*(S, x, \sigma_{n+1}).$$

By the inductive hypothesis,

$$\eta^*(S, x) = \{(q_n, I(q_0) \land \delta(q_0, \sigma_1, q_1) \land \cdots \land \delta(q_{n-1}, \sigma_n, q_n)) : q_0, \ldots, q_n \in Q$ and $r_n = I(q_0) \land \delta(q_0, \sigma_1, q_1) \land \cdots \land \delta(q_{n-1}, \sigma_n, q_n) \neq 0\}. $$

By the definition of $\eta$,

$$\eta^*(S, x, \sigma_{n+1}) = \bigcup_{(q_n, r_n) \in \eta^*(S, x, \sigma_{n+1})} \{(q_{n+1}, r_n \land \delta(q_n, \sigma_{n+1}, q_{n+1})) : q_{n+1} \in Q$ and $r_n \land \delta(q_n, \sigma_{n+1}, q_{n+1}) \neq 0\} = \{((q_{n+1}, I(q_0) \land \delta(q_0, \sigma_1, q_1) \land \cdots \land \delta(q_{n-1}, \sigma_n, q_n) \land \delta(q_n, \sigma_{n+1}, q_{n+1})) : q_0, \ldots, q_{n+1} \in Q$ and $r_{n+1} = I(q_0) \land \delta(q_0, \sigma_1, q_1) \land \cdots \land \delta(q_{n-1}, \sigma_n, q_n) \land \delta(q_n, \sigma_{n+1}, q_{n+1}) \neq 0\}$$

which establishes the inductive hypothesis.

By the definition of $l$-valued final state $E$, for any input $\omega = \sigma_1 \cdots \sigma_n \in \Sigma^*(n \geq 0)$, we have

$$[\text{rec}_\mathcal{A}^d(\omega)] = E(\eta^*(S, \omega)) = \bigvee \{r \land F(q) : (q_n, r_n) \in \eta^*(S, \omega)\} = \bigvee \{I(q_0) \land \delta(q_0, \sigma_1, q_1) \land \cdots \land \delta(q_{n-1}, \sigma_n, q_n) \land F(q_n) : q_0, \ldots, q_n \in Q$ and $I(q_0) \land \delta(q_0, \sigma_1, q_1) \land \cdots \land \delta(q_{n-1}, \sigma_n, q_n) \neq 0\}.$$


0) = \sqrt{\{I(q_0) \land \delta(q_0, \sigma_1, q_1) \land \cdots \land \delta(q_{n-1}, \sigma_n, q_n) \land F(q_n) : q_0, \cdots, q_n \in Q\}} = [rec_{\mathcal{A}}(\omega)]. \text{ Thus } rec_{\mathcal{A}^d} = rec_{\mathcal{A}}, \mathcal{A} \text{ and } \mathcal{A}^d \text{ are equivalent.} \tag*{\Box}

Theorem 2.1 gives the subset construction of finite automaton in the frame of quantum logic. In fact, in the case of \( l = \{0, 1\} \), the underlying logic is the classical logic, the quantum subset construction is just the ordinary subset construction.

We give an example to illustrate the technique of the quantum subset construction introduced above.

**Example 2.1.** Let \( \otimes^2 \mathbb{C}^2 \) be the 2-qubit state space, where \( \mathbb{C} \) denotes the set of complex numbers. All the closed subspaces of Hilbert space \( \otimes^2 \mathbb{C}^2 \), denoted by \( l \), forms a (complete) orthomodular lattice (5), \( (l, \leq, \land, \lor, \bot, 0, 1) \), with usual notations. As the standard notation in quantum computation (14, 12), \( |0 \rangle, |1 \rangle \) are four basis states in the 2-qubit state space. We use \( a_{ij} = \text{span}(|i \rangle | j \rangle) \) to denote the closed subspace spanned by \( |i \rangle | j \rangle \), \( i, j = 0, 1 \).

An \( l \)-VFA \( \mathcal{A} = (Q, \Sigma, \delta, I, F) \) is defined as follows (c.f., 14), \( Q = \{p, q\} \), \( \Sigma = \{\sigma\} \), \( I(p) = 1 \) and \( I(q) = a_{10} \), \( F(p) = a_{10} \) and \( F(q) = 1 \), and \( \delta(p, \sigma, q) = a_{00}, \delta(p, \sigma, p) = a_{01}, \delta(q, \sigma, q) = a_{10} \) and \( \delta(q, \sigma, p) = a_{11} \).

Using the quantum subset construction, the determinization of \( \mathcal{A} \) is induced as follows. In this example, \( l \) is an infinite orthomodular lattice, and \( l_1 = \{a_{00}, a_{01, a_{10}, a_{11}, 0, 1}\} \). In the construction of \( \mathcal{A}^d \), the state set \( Q^d \) is \( 2^{Q \times \{l_1 - \{0\}\}} \), \( \mathcal{A}^d \) will have \( 2^{10} \) states. To give a full construction of \( \mathcal{A}^d \) is a tedious work. However, it is sufficient to give those states which are useful in generating the \( l \)-valued language recognized by \( \mathcal{A}^d \) from the initial state \( S \).

The initial state is \( S = \{(p, 1), (q, a_{10})\} \). By the simple calculation, we have \( \eta(S, \sigma) = \{(p, a_{01}), (q, a_{00}), (q, a_{10})\} \), \( \eta(S, \sigma) = \{(p, a_{01}), (q, a_{00}), (q, a_{10})\} \), and \( \eta(S, \sigma) = \{(p, a_{01}), (q, a_{00}), (q, a_{10})\} \). Therefore, the useful states of \( \mathcal{A}^d \) are \( S \), \( \{(p, a_{01}), (q, a_{00}), (q, a_{10})\} \) and \( \{(p, a_{01}), (q, a_{10})\} \), which are denoted as \( p_0, p_1 \) and \( p_2 \) respectively. Let \( P = \{p_0, p_1, p_2\} \), then the state transition function \( \eta \) is defined as, \( \eta(p_0, \sigma) = p_1, \eta(p_1, \sigma) = p_2 \) and \( \eta(p_2, \sigma) = p_2 \).

The \( l \)-valued final state \( E \) is defined as, \( E(p_0) = (1 \land F(p)) \lor (a_{10} \land F(q)) = a_{01} \lor a_{10}, E(p_1) = (a_{01} \land F(p)) \lor (a_{00} \land F(q)) \lor (a_{10} \land F(q)) = a_{01} \lor a_{00} \lor a_{10} \) and \( E(p_2) = (a_{01} \land F(p)) \lor (a_{10} \land F(q)) = a_{01} \lor a_{10} \). This complete the construction of \( \mathcal{A}^d = (P, \Sigma, \eta, p_0, E) \). Then \( rec_{\mathcal{A}}(= rec_{\mathcal{A}^d}) \) can be simply calculated as follows,

\[
[rec_{\mathcal{A}}(\omega)] = \begin{cases} 
  a_{01} \lor a_{00} \lor a_{10}, & \text{if } \omega = \sigma, \\
  a_{01} \lor a_{10}, & \text{otherwise.}
\end{cases}
\]
We continue to study the relationship between \( l \)-VFA and \( l \)-VFA with \( \varepsilon \)-moves. Let us first recall the definition of \( l \)-VFA with \( \varepsilon \)-moves.

**Definition 2.3.** \[19\] An \( \varepsilon \)-moves \( l \)-valued automaton (\( l \)-VFA\( _\varepsilon \) for short) is a five-tuple \( \mathcal{A} = (Q, \Sigma, \delta, I, F) \) in which all components are the same as in an \( l \)-valued automaton (without \( \varepsilon \)-moves), but the domain of the quantum transition relation \( \delta \) is changed to \( Q \times (\Sigma \cup \{\varepsilon\}) \times Q \); that is, \( \delta \) is a mapping from \( Q \times (\Sigma \cup \{\varepsilon\}) \times Q \) into \( l \), where \( \varepsilon \) stands for the empty string of input symbols.

Now let \( \mathcal{A} = (Q, \Sigma, \delta, I, F) \) be an \( l \)-valued automaton with \( \varepsilon \)-moves. Then the recognizability \( rec_\mathcal{A} \) is also defined as an \( l \)-valued unary predicate over \( \Sigma^* \), and it is given by

\[
rec_\mathcal{A}(\omega) = (\exists n \geq 0)(\exists \tau_1 \in \Sigma \cup \{\varepsilon\}) \cdots (\exists \tau_n \in \Sigma \cup \{\varepsilon\}).(\exists q_0 \in Q) \cdots (\exists q_n \in Q).(q_0 \in I \land q_n \in F \land (q_0, \tau_1, q_1) \in \delta \land \cdots \land (q_{n-1}, \tau_n, q_n) \in \delta \land \tau_1 \cdots \tau_n = \omega)
\]

for all \( \omega \in \Sigma^* \). The defining equation of \( rec_\mathcal{A} \) may be rewritten in terms of truth value as follows:

\[
[rec_\mathcal{A}(\omega)] = \bigvee \{I(q_0) \land \delta(q_0, \tau_1, q_1) \land \cdots \land \delta(q_{n-1}, \tau_n, q_n) \land F(q_n) : n \geq 0, \tau_1, \cdots, \tau_n \in \Sigma \cup \{\varepsilon\} \text{ satisfying } \tau_1 \cdots \tau_n = \omega, \text{ and } q_0, \cdots, q_n \in Q\}.
\]

We shall show that \( l \)-VFA and \( l \)-VFA\( _\varepsilon \) are equivalent in the sequel. First, we study a special kind of \( l \)-VFA\( _\varepsilon \) in which quantum transition is crisp, that is, \( \delta \) is a crisp subset of \( Q \times (\Sigma \cup \{\varepsilon\}) \times Q \). In this case, \( \delta \) can be seen as a mapping from \( Q \times (\Sigma \cup \{\varepsilon\}) \) to \( 2^Q \).

Let \( \mathcal{A} = (Q, \Sigma, \delta, q_0, F) \) be an \( l \)-VFA\( _\varepsilon \) with crisp quantum transition and with a unique initial state \( q_0 \in Q \), the explicit expression of \( rec_\mathcal{A} \) can be induced as follows. First, we give the extension \( \delta^* : 2^Q \times \Sigma^* \to 2^Q \) using the notion of \( \varepsilon \)-closure. For \( q \in Q \), the \( \varepsilon \)-closure of \( q \), denoted \( EC(q) \), is defined as,

\[
EC(q) = \{ p \in Q : \text{there exists } n \geq 0 \text{ and } q_0, \cdots, q_n \text{ satisfying } q_i \in \delta(q_{i-1}, \varepsilon) \text{ for any } i = 1, \cdots, n, \text{ in which } q_0 = q \text{ and } q_n = p\}.
\]

For any subset \( X \) of \( Q \), the \( \varepsilon \)-closure of \( X \), denoted \( EC(X) \), is defined as

\[
EC(X) = \bigcup_{q \in X} EC(q).
\]

In particular, \( EC(\{q\}) = EC(q) \). Then \( \delta^* \) is defined inductively as,

\[
\delta^*(q, \varepsilon) = EC(q),
\]

\[
\delta^*(q, \omega \sigma) = EC(\delta(\delta^*(q, \omega), \sigma)) \text{ for any } q \in Q, \omega \in \Sigma^* \text{ and } \sigma \in \Sigma.
\]

Then

\[
\delta^*(X, \omega) = \bigcup_{q \in X} \delta^*(q, \omega).
\]

It follows that

\[
\delta^*(q, \omega \sigma) = \delta^*(\delta^*(q, \omega), \sigma)
\]
for any \( q \in Q, \omega \in \Sigma^* \) and \( \sigma \in \Sigma \). By the definition of unitary predicate \( \text{rec} \) over \( \Sigma^* \), the truth valued \( \text{rec}_A \) for an \( l\)-VFA with crisp quantum transition is defined as follows: for any \( \omega \in \Sigma^* \),

\[
[\text{rec}_A](\omega) = \bigvee \{ F(q) : q \in \delta^*(q_0, \omega) \}.
\]

We construct an equivalent \( l\)-VFA \( B \) from the above \( A \) as follows, where \( B = (Q, \Sigma, \eta, q_0, E) \).

The quantum transition \( \eta \) is defined as: for any \( q \in Q \) and \( \sigma \in \Sigma \),

\[
\eta(q, \sigma) = \delta^*(q, \sigma).
\]

If \( q \neq q_0 \), then

\[
E(q) = F(q),
\]

and

\[
E(q_0) = \bigvee \{ F(q) : q \in \mathcal{E}(q_0) \}.
\]

Note that \( B \) has no \( \varepsilon \)-transitions.

**Lemma 2.2.** For any \( l\)-VFA with crisp quantum transition \( A \), the \( l\)-VFA \( B \) constructed as above is equivalent to \( A \), i.e., \( \text{rec}_A = \text{rec}_B \).

**Proof** We wish to show by induction on \( |\omega| \) that \( \eta^*(q, \omega) = \delta^*(q, \omega) \). However, this statement may not be true for \( \omega = \varepsilon \), since \( \eta^*(q, \varepsilon) = \{q\} \), while \( \delta^*(q, \varepsilon) = \mathcal{E}(q) \). We therefore begin our induction at 1.

Let \( |\omega| = 1 \). Then \( \omega \) is a symbol \( \sigma \), and \( \eta(q, \sigma) = \delta^*(q, \sigma) \) by definition of \( \eta \). Suppose that the hypothesis holds for inputs of length \( n \) or less. Let \( \omega = x\sigma \) be a string of length of \( n + 1 \) with symbol \( \sigma \) in \( \Sigma \). Then

\[
\eta^*(q, x\sigma) = \eta(\eta^*(q, x), \sigma).
\]

By the inductive hypothesis, \( \eta^*(q, x) = \delta^*(q, x) \). Let \( \delta^*(q, x) = X \), we must show that \( \eta(X, \sigma) = \delta^*(q, x\sigma) \). But

\[
\eta(X, \sigma) = \bigcup_{q \in Q} \eta(q, \sigma) = \bigcup_{q \in X} \delta^*(q, \sigma).
\]

Then as \( X = \delta^*(q, x) \) we have

\[
\bigcup_{q \in X} \delta^*(q, \sigma) = \delta^*(q, x\sigma).
\]

Thus

\[
\eta^*(q, x\sigma) = \delta^*(q, x\sigma).
\]

To complete the proof we shall show that \( [\text{rec}_B(\omega)] = \bigvee \{ F(q) : q \in \delta^*(q_0, \omega) \} \).

If \( \omega = \varepsilon \), this statement is immediate from the definition of \( E \). That is, \( \eta^*(q_0, \varepsilon) = \{q_0\} \), then \( [\text{rec}_B(\varepsilon)] = \bigvee \{ E(q) : q \in \eta^*(q_0, \varepsilon) \} = E(q_0) = \bigvee \{ F(q) : q \in \delta^*(q_0, \varepsilon) \} \).

If \( \omega \neq \varepsilon \), then \( \omega = x\sigma \) for some symbol \( \sigma \). We have two cases to discuss.

Case I: \( q_0 \notin \eta^*(q_0, x\sigma) \). By the definition of \( E \) and the equality \( \eta^*(q_0, x\sigma) = \delta^*(q_0, x\sigma) \), it follows that
δ

∩

rec

A

with crisp quantum transition from

q

∈

q, r

for any

construction from an

l

is defined by ,

for any

Z

equivalence between

Lemma 2.3.

For any

Theorem 2.2.

Proof

rec

i.e.,

A

= A

rec

rec

with crisp quantum transition from

A

as follows.

Let

A = (Q, Σ, δ, I, F )

be an l-VFAε.

We construct an equivalent l-VFAε

B = (P, Σ, η, S, E )

with crisp quantum transition from

A

as follows.

Let

X = Im(δ)∪Im(I)∪Im(F ),

and

l1 = XA.

Choose

P = 2Q×(l1−{0}) ,

and

S = \{(q, I(q)) : q ∈ Q and I(q) ≠ 0\},

then

P

is a finite set and

S

∈

P .

The state transition

η : P×(Σ∪{ε}) → P

defined by,

η(\{(q, r)\}, τ) = \{(p, r ∩ (q, τ, p)) : p ∈ Q and r ∩ (q, τ, p) ≠ 0\}

for any

(q, r) ∈ Q × (l1 − {0})

and

τ ∈ Σ∪{ε}. We define

η(Z, τ) = \bigcup_{(q, r) ∈ Z} η(\{(q, r)\})

for any

Z

∈

P

and

τ

∈

Σ∪{ε}. Then

η

is well defined as discussed in the quantum subset

collection from an l-VFA to an l-VDFA. The quantum final state

E : P → l

defined as,

E(Z) = \bigvee \{r ∩ F(q) : (q, r) ∈ Z\}.

Lemma 2.3. For any l-VFAε

A = (Q, Σ, δ, I, F ),

the l-VFAε with crisp quantum transition

B

constructed as above is equivalent to

A,

i.e.,

recA = recB.

Proof

The proof is very similar to that of Theorem 2.1,

we omit it here.

Combining the above two lemmas,

we can conclude the following theorem which shows the

equivalence between l-VFAε and l-VFA.

Theorem 2.2. For any l-VFAε

A ,

there is an l-VFA

B

such that

A

and

B

are equivalent, i.e.,

recA = recB.

Combining Theorem 2.1 and Theorem 2.2,

we can see the equivalence between l-VFAε,

l-VFA and l-VDFA.

Corollary 2.1. For any l-VFAε

A ,

there is an l-VDFA

B

such that

A

and

B

are equivalent, i.e.,

recA = recB.

As an application of Theorem 2.1,

we present pumping lemma in the frame of quantum

logic as follows.
**Proposition 2.2.** (Pumping lemma in quantum logic) For an l-regular language $A : \Sigma^* \to l$, there exists positive integer $n$, for any input string $z \in \Sigma^*$, if $|z| \geq n$, then there are $u, v, w \in \Sigma^*$ such that $|uv| \leq n$, $v \neq \varepsilon$, $z = uvw$, and for any non-negative integer $l$, the equality $A(uv^lw) = A(uvw)$ holds.

**Proof** Since $A$ is l-regular, it is accepted by an l-VFDA $A = (Q, \Sigma, \delta, q_0, F)$ with some particular number of states, say $n$. Consider an input of $n$ or more symbols $z = \sigma_1 \cdots \sigma_m$, $m \geq n$, and for $i = 1, \cdots, m$, let $\delta^*(q_0, \sigma_1 \cdots \sigma_i) = q_i$. It is not possible for each of the $n+1$ states $q_0, \cdots, q_n$ be different, since there are only $n$ different states. Thus there are two integers $j$ and $k$, $0 \leq j < k \leq n$, such that $q_j = q_k$. Let $u = \sigma_1 \cdots \sigma_j$, $v = \sigma_{j+1} \cdots \sigma_k$, $w = \sigma_{k+1} \cdots \sigma_m$, then $|uv| = k \leq n$, $v \neq \varepsilon$ and $z = uvw$. Observing that $\delta^*(q_0, \sigma_1 \cdots \sigma_j) = \delta^*(\delta^*(q_0, \sigma_1 \cdots \sigma_j), \sigma_{j+1} \cdots \sigma_m) = \delta^*(q_j, \sigma_{j+1} \cdots \sigma_m) = \delta^*(q_k, \sigma_{k+1} \cdots \sigma_m) = q_m$, and for any $l \geq 1$, $\delta^*(q_0, \sigma_1 \cdots \sigma_j \sigma_{j+1} \cdots \sigma_k) = \delta^*(\delta^*(q_0, \sigma_1 \cdots \sigma_j), \sigma_{j+1} \cdots \sigma_k) = \delta^*(q_j, \sigma_{j+1} \cdots \sigma_k) = \delta^*(q_k, \sigma_{k+1} \cdots \sigma_m) = q_m$. Therefore, for any $l \geq 0$, $A(uvw^lw) = [\text{rec}_A(uvw^lw)] = F(\delta^*(q_0, uv^lw)) = F(q_m) = F(\delta^*(q_0, uvw)) = [\text{rec}_A(uvw)] = A(uvw)$. \hfill \Box

**Remark 2.1.** Lemma 2.2, Lemma 2.3, Theorem 2.2, Corollary 2.1 and Proposition 2.2 (and all propositions in Section 3) can be restated in the language of quantum logic, as done in Theorem 2.1; we left them to the readers which are interested in stating the related propositions in logic language.

### 3 Kleene Theorem for l-valued finite automata

We use $lR(\Sigma)$ to denote the set of l-regular languages over $\Sigma$. Up to now, we still do not know whether $lR(\Sigma)$ is closed under the operations of meet, complement and Kleene closure of l-valued regular languages. Indeed, in [19], Ying gave some conditions using the notion of commutators to guarantee $lR(\Sigma)$ being closed under the above mentioned operations. Since the above mentioned restrictions, Kleene theorem for l-VFA depends on the notion of commutators. We shall show that all these restrictions are not necessary in this section. In fact, we shall show that $lR(\Sigma)$ is closed under the operations of meet, complement and Kleene closure of l-valued regular languages. Furthermore, Kleene theorem holds in the frame of quantum logic.

Let us recall the operations of l-valued languages ([19]): for $A, B \in l(\Sigma^*)$ and $r \in l$, the union $A \lor B$, the intersection $A \land B$, the complement $A^\perp$, the scalar product $rA$, the concatenation $AB$, the Kleene closure $A^*$ are defined as follows: for any $\omega \in \Sigma^*$, $A \lor B(\omega) = A(\omega) \lor B(\omega)$,
Corollary 3.1. Let $A : \Sigma^* \to l$ be an $l$-valued language over $\Sigma$. Then the following statements are equivalent.

1. $A$ is an $l$-regular language.
2. There exist $k_1, \ldots, k_m \in l - \{0\}$, and regular languages $L_1, \ldots, L_m$ such that $A = \bigvee_{i=1}^m k_i 1_{L_i}$, where $1_{L_i}$ denotes the characteristic function of $L_i$.
3. There exist $k_1, \ldots, k_m \in l - \{0\}$, and pairwise disjoint regular languages $L_1, \ldots, L_m$ satisfying the equality $A = \bigvee_{i=1}^m k_i 1_{L_i}$.

Proof (1) $\implies$ (3) Since $A$ is an $l$-valued regular language, there is an $l$-V DFA $A = (Q, \Sigma, \delta, q_0, F)$ recognized $A$. That is, for all $\omega \in \Sigma^*$, $A(\omega) = \lceil \text{rec}_A(\omega) \rceil = F(\delta^*(q_0, \omega))$. Write $\text{Im}(F) - \{0\} = \{k_1, \ldots, k_m\}$, and let $F_i = \{q \in Q : F(q) = k_i\}$. For this $F_i$, we construct a DFA, $A_i = (Q, \Sigma, \delta, q_0, F_i)$. Let the language recognized by $A_i$ be $L_i$, then $L_i$ is a regular language, and evidently, the family $\{L_1, \ldots, L_m\}$ is pairwise disjoint. Moreover, $A(\omega) = r$ iff $F(\delta^*(q_0, \omega)) = r$, iff there is $i$ such that $r = k_i$ and $\omega \in L_i$, which shows that $A = \bigvee_{i=1}^m k_i 1_{L_i}$.

(3) $\implies$ (2) is obvious.

(2) $\implies$ (1) Since each $L_i$ is regular, there is a DFA $A_i = (Q_i, \Sigma, \delta, q_0, F_i)$ recognized $L_i$. We can assume that $Q_i \cap Q_j = \emptyset$ whenever $i \neq j$. Define an $l$-VFA, $A = (Q, \Sigma, \delta, q_0, F)$ as follows, $Q = \bigcup_{i=1}^m Q_i \cup \{q_0\}$, where $q_0 \notin \bigcup_{i=1}^m Q_i$, and $\delta : Q \times \Sigma \to 2^Q$ is, $\delta(q, \sigma) = \{\delta_1(q_01, \sigma), \ldots, \delta_m(q_0m, \sigma)\}$, for $q \in Q_i$, $\delta(q, \sigma) = \delta_i(q, \sigma)$; $F(q_0) = \bigvee \{k_i : q_0i \in F_i\}$, and when $q \neq q_0$,

$$F(q) = \begin{cases} k_i, & \text{if } q \in F_i \\ 0, & \text{otherwise.} \end{cases}$$

Then it can be easily verified that $A = \text{rec}_A = \bigvee_{i=1}^m k_i 1_{L_i}$. Hence $A$ is an $l$-valued regular language.

We call the $l$-valued language satisfying the condition (2) or (3) in the above theorem the $l$-valued recognizable step language, and write the set of all $l$-valued recognizable languages on $\Sigma$ as $\text{step}(\Sigma)$, which is equal to $lR(\Sigma)$.

The following proposition gives the level characterization of $l$-valued recognizable step languages.

Corollary 3.1. Let $A : \Sigma^* \to l$ be an $l$-valued language over $\Sigma$. Then the following statements are equivalent.

1. $A$ is an $l$-regular language.
Theorem 3.2. The family $\text{step}(\Sigma)$ or $lR(\Sigma)$ is closed under the operations of union, intersection, scalar product, complement, concatenation and Kleene closure.

Proof Let $A, B \in \text{step}(\Sigma)$. By Theorem 3.1, we can assume $A = \bigvee_{i=1}^{m} k_i 1_{L_i}$, $B = \bigvee_{j=1}^{n} d_j 1_{M_j}$, where, all $L_i$ and $M_j$ are regular languages and $\{L_i\}_{i=1}^{m}$ are pairwise disjoint, $\{M_j\}_{j=1}^{n}$ are also pairwise disjoint.

With respect to the union, we have $A \lor B = \bigvee_{i=1}^{m} k_i 1_{L_i} \lor \bigvee_{j=1}^{n} d_j 1_{M_j}$. By Theorem 3.1, it follows that $A \lor B \in \text{step}(\Sigma)$.

With respect to the intersection, we have $A \land B = \bigvee_{i=1}^{m} \bigwedge_{j=1}^{n} (k_i \land d_j) 1_{L_i \cap M_j}$. By Theorem 3.1, it follows that $A \land B \in \text{step}(\Sigma)$.

With respect to the scalar product, for each $r \in l$, we have $rA(\omega) = r \land A(\omega)$, then $rA = \bigvee_{i=1}^{m} (r \land k_i) 1_{L_i}$. Therefore, $rA \in \text{step}(\Sigma)$.

For the complement operation, since $A^\perp(\omega) = A(\omega)^\perp$, it follows that $A^\perp = \bigvee_{i=1}^{m} k_i^\perp 1_{L_i} \lor 1_{\Sigma^* - (L_1 \cup \cdots \cup L_m)}$. By Theorem 3.1, it follows that $A^\perp \in \text{step}(\Sigma)$.

For the operation of concatenation, since $AB(\omega) = \bigvee \{A(\omega_1) \land B(\omega_2) : \omega = \omega_1\omega_2\}$, it follows that $AB = \bigvee_{i=1}^{m} \bigwedge_{j=1}^{n} (k_i \land d_j) 1_{L_iM_j}$. This shows that $AB \in \text{step}(\Sigma)$.

For the Kleene closure, $A^*$ is defined by, $A^*(\omega) = \bigvee \{A(\omega_1) \land \cdots \land A(\omega_k) : k \geq 0, \omega = \omega_1 \cdots \omega_k\}$ for any $\omega \in \Sigma^*$. Since $A = \bigvee_{i=1}^{m} k_i 1_{L_i}$, and $L_1, \ldots, L_m$ are pairwise disjoint regular languages and $k_i \neq 0$ for each $i$, it follows that $\text{Im}(A) - \{0\} = \{k_1, \ldots, k_m\}$, and $L_i = \{\omega \in \Sigma^* : A(\omega) = k_i\}$ ($i = 1, \ldots, m$). For any nonempty subset $K$ of the set $\{1, 2, \ldots, m\}$, we can assume that $K = \{i_1, \ldots, i_s\}$. Let $r_K = r_{i_1} \land \cdots \land r_{i_s}$, $L(K) = \bigcup_{p_1 \cdots p_s} L_{p_1}^L L_{p_2}^L L_{p_3}^L (L_{p_1} \cup L_{p_2})^* \cdots L_{p_{s-1}}^L (L_{p_1} \cup \cdots \cup L_{p_{s-2}}) L_{p_s}^L (L_{p_1} \cup \cdots \cup L_{p_s})^*$, where $p_1 \cdots p_s$ is a permutation of $\{i_1, \ldots, i_s\}$, and $L(K)$ is taken unions under all permutations of $\{i_1, \ldots, i_s\}$. Hence $L(K)$ is a regular language. It is easily verified that $A^* = \bigvee_{\emptyset \neq K \subseteq \{1, 2, \ldots, m\}} r_K 1_{L(K)} \lor 1_{\{\varepsilon\}}$. By Theorem 3.1, it follows that $A^* \in \text{step}(\Sigma)$.

Definition 3.1. The language of $l$-valued regular expressions over alphabet $\Sigma$ has the alphabet $(\Sigma \cup \{\varepsilon, \emptyset\}) \cup (l \cup \{+, \cdot, *\})$. The symbols in $\Sigma \cup \{\varepsilon, \emptyset\}$ will be used to denote atomic expressions, and the symbols in $l \cup \{+, \cdot, *\}$ will be used to stand for operators for building up compound expressions: $*$ and all $r \in l$ are the unary operators, and $+, \cdot$ are binary ones. We use $\alpha, \beta$ to act as meta-symbols for regular expressions and $L(\alpha)$ for the language denoted by
expression $\alpha$. More explicitly, $L(\alpha)$ will be used to denote an $l$-valued subset of $\Sigma^*$; that is, $L(\alpha) \in l^{\Sigma^*}$. The $l$-valued regular expressions and the $l$-valued languages denoted by them are formally defined as follows:

1. For each $\sigma \in \Sigma$, $\sigma$ is a regular expression, and $L(\sigma) = \{\sigma\}$; $\varepsilon$ and $\emptyset$ are regular expressions, and $L(\varepsilon) = \{\varepsilon\}$, $L(\emptyset) = \emptyset$.

2. If both $\alpha$ and $\beta$ are regular expressions, then for each $r \in l$, $r\alpha$, $\alpha + \beta$, $\alpha \cdot \beta$, $\alpha^*$ are all regular expressions, and $L(r\alpha) = rL(\alpha)$, $L(\alpha + \beta) = L(\alpha) \lor L(\beta)$, $L(\alpha \cdot \beta) = L(\alpha)L(\beta)$, $L(\alpha^*) = L(\alpha)^*$.

**Theorem 3.3.** (Kleene Theorem in quantum logic) For an $l$-valued language $A \in l(\Sigma^*)$, $A$ can be recognized by an $l$-VFA iff there exists an $l$-valued regular expression $\alpha$ over $\Sigma$ such that $A = L(\alpha)$.

**Proof** If $A$ can be recognized by an $l$-VFA, then by Theorem 3.1, there exist $k_1, \ldots, k_n \in l - \{0\}$, and regular languages $L_1, \ldots, L_n$ such that $A = \bigvee_{i=1}^n k_iL_i$. Since each $L_i$ is a regular language, by classical Kleene Theorem, there exists a regular expression $\alpha_i$ over $\Sigma$ such that $L(\alpha_i) = L_i$. Let $\alpha = k_1\alpha_1 + \cdots + k_n\alpha_n$, then $\alpha$ is an $l$-valued regular expression, and $L(\alpha) = \bigvee_{i=1}^n k_iL(\alpha_i) = \bigvee_{i=1}^n k_iL_i = A$.

Conversely, assume that there exists an $l$-valued regular expression $\alpha$ such that $A = L(\alpha)$. We show that $A$ can be recognized by an $l$-VFA inductively on the number of operation symbols occurring in $\alpha$. If there is no operation symbol in $\alpha$, then $\alpha = \sigma \in \Sigma, \varepsilon$ or $\emptyset$. In this case, $L(\alpha) = \{\sigma\}, \{\varepsilon\}$ or $\emptyset$, and $L(\alpha)$ can be recognized by a classical DFA. The classical DFA is evidently an $l$-VDFA, so $L(\alpha)$ can be recognized by an $l$-VDFA in this case. Inductively, since the family of recognizable languages by $l$-VDFA is closed under union, intersection, scalar product, concatenation and Kleene closure (by Theorem 3.2), it follows that $L(\alpha)$ can be recognized by an $l$-VDFA for any $l$-valued regular expression $\alpha$. \qed

### 4 Conclusion

In this paper, we introduced the quantum subset construction of orthomodular lattice-valued finite automata, then we proved the equivalence between orthomodular lattice-valued finite automata, orthomodular lattice-valued deterministic finite automata and orthomodular lattice-valued finite automata with $\varepsilon$-moves. We give a simple characterization of orthomodular lattice-valued languages recognized by orthomodular lattice-valued finite automata, then we proved that the Kleene theorem holds in the frame of quantum logic, many results in [19] can be
strengthen such as the pumping lemma in the frame of quantum logic using the results of this paper.

References

[1] A. Ambainis, J. Watrous, Two-way finite automata with quantum and classical states, Theoretical Computer Science, 287(2002), 299-311.
[2] S. Eilenberg, Automata, Languages and Machines, vol. A, vol B, Academic Press, New York, 1974.
[3] Z. Ésik, W. Kuich, Modern Automata Theory, 2007, see [http://dmg.tuwien.ac.at/kuich/](http://dmg.tuwien.ac.at/kuich/).
[4] J. Gruska, Quantum Computing, McGraw-Hill, London, 1999.
[5] J. E. Hopcroft, J. D. Ullman, Introduction to Automata Theory, Languages and Computation, Addison-Wesley, New York, 1979.
[6] G. Kalmbach, Orthomodular Lattices, Academic Press, London, 1983.
[7] B. Khoussainov, A. Nerode, Automata Theory and its Applications, Birkäuser, Boston, 2001.
[8] S. C. Kleene, Representation of events in nerve nets and finite automata, in: Automata Studies, ed. by C. E. Shannon and J. McCarthy, Princeton University Press, Princeton, NJ, 1956, 3-42.
[9] Y. M. Li, Z. H. Li, Free semilattices and strongly free semilattices generated by partially ordered sets, Northeastern Mathematical Journal, 9(3)(1993), 359-366.
[10] Y. M. Li, W. Pedrycz, Fuzzy finite automata and fuzzy regular expressions with membership values in lattice-ordered monoids, Fuzzy Sets and Systems, 156(2005), 68-92.
[11] C. Moore, J. P. Crutchfield, Quantum automata and quantum grammars, Theoretical Computer Science, 237(2000), 275-306.
[12] M. A. Nielsen, I. L. Chuang, Quantum Computation and Quantum Information, Cambridge University, Cambridge, 2000.
[13] D. W. Qiu, Automata theory based on quantum logic: some characterizations, Information and Computation, 190(2004), 179-195.
[14] D. W. Qiu, Automata theory based on quantum logic: reversibilities and pushdown automata, Theoretical Computer Science, 386(2007), 38-56.
[15] D. W. Qiu, Notes on automata theory based on quantum logic, Science in China Series F: Information Sciences, 50(2)(2007), 154-169.
[16] M. O. Rabin, D. Scott, Finite automata and their decision problems, IBM J. Research and Development, 3(1959), 114-125.
[17] M.S. Ying, Automata theory based on quantum logic (I), International Journal of Theoretical Physics, 39(2000), 981-991.

[18] M.S. Ying, Automata theory based on quantum logic (II), International Journal of Theoretical Physics, 39(2000), 2545-2557.

[19] M.S. Ying, A theory of computation based on quantum logic (I), Theoretical Computer Science, 344(2005), 134-207.