Quantum Speed-ups for Single-machine Scheduling Problems

Camille Grange, Eric Bourreau, Michael Poss, Vincent t'Kindt

Quantum Speed-ups for Single-machine Scheduling Problems

Camille Grange
Laboratory of Computer Science, Robotics and Microelectronics of Montpellier
France
camille.grange@lirmm.fr

Eric Bourreau
Laboratory of Computer Science, Robotics and Microelectronics of Montpellier
France
eric.bourreau@lirmm.fr

Michael Poss
Laboratory of Computer Science, Robotics and Microelectronics of Montpellier
France
michael.poss@lirmm.fr

Vincent T'Kindt
Laboratory of Fundamental and Applied Computer Science of Tours
France
tkindt@univ-tours.fr

ABSTRACT

Grover search is currently one of the main approaches to obtain quantum speed-ups for combinatorial optimization problems. The combination of Quantum Minimum Finding (obtained from Grover search) with dynamic programming has proved particularly efficient to improve the worst-case complexity of several NP-hard optimization problems. Specifically, for these problems, the classical dynamic programming complexity (ignoring the polynomial factors) in $O^*(c^n)$ can be reduced by a bounded-error hybrid quantum-classical algorithm to $O^*(c^n_{quant})$ for $c_{quant} < c$. In this paper, we extend the resulting hybrid dynamic programming algorithm to three examples of single-machine scheduling problems: minimizing the total weighted completion time with deadlines, minimizing the total weighted completion time with precedence constraints, and minimizing the total weighted tardiness. The extension relies on the inclusion of a pseudo-polynomial term in the state space of the dynamic programming as well as an additive term in the recurrence.

CCS CONCEPTS

• **Theory of computation → Design and analysis of algorithms**; • **Hardware → Quantum computation**; • **Mathematics of computing → Combinatorial optimization**.

KEYWORDS

Quantum optimization, Grover search, scheduling, dynamic programming

ACM Reference Format:
Camille Grange, Eric Bourreau, Michael Poss, and Vincent T'Kindt. 2023. Quantum Speed-ups for Single-machine Scheduling Problems. In *Genetic and Evolutionary Computation Conference Companion (GECCO '23 Companion), July 15–19, 2023, Lisbon, Portugal.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3583133.3596415

GECCO '23 Companion, July 15–19, 2023, Lisbon, Portugal

ACM ISBN 979-8-4007-0120-7/23/07…$15.00
https://doi.org/10.1145/3583133.3596415

1 INTRODUCTION

The interest in quantum computing to solve combinatorial optimization problems has been growing for several years in the operational research community. More precisely, two branches are distinguished. The first one relates to heuristics, often hybrid quantum-classical algorithms such as variational quantum algorithms [3, 8] and in particular QAOA [6]. Essentially, these algorithms require the optimization problem to be formulated as a QUBO (Quadratic Unconstrained Binary Optimization) and can be implemented on current noisy quantum computers because the quantum part can be made rather small. The second branch relates to *exact* algorithms. Unlike the previous algorithms, it is impossible to implement them today but theoretical speed-ups have been proved for several types of problems and algorithms [13, 16].

The most emblematic algorithm of this branch is Grover search [9], which achieves a quadratic speed-up when searching for a specific element in an unsorted table, where the complexity is computed as the number of queries of the table and done by an oracle. The authors of [5] use Grover search as a subroutine for a hybrid quantum-classical algorithm that finds with high probability the minimum of an unsorted table, leading to the algorithm known as Quantum Minimum Finding (QMF). Later, the authors of [2] combine QMF with dynamic programming to address NP-hard optimization problems. They apply their algorithm to vertex ordering problems, the Traveling Salesman Problem (TSP), and the Minimum Set Cover problem, among others. All these problems satisfy a specific property which implies that they can be solved by classical dynamic programming in $O^*(c^n)$, where O^* is the usual asymptotic notation that ignores the polynomial factors, and c is usually not smaller than 2. The hybrid algorithm from [2] reduces the complexity to $O^*(c^n_{\text{quant}})$ for $c_{\text{quant}} < c$. For instance, the TSP is solved by Held and Karp dynamic programming [10] in $O^*(2^n)$, and by the hybrid algorithm of [2] in $O^*(1.728^n)$. Subsequently to the work of [2], other NP-hard problems have been tackled with the idea of combining Grover search (or QMF) and classical dynamic programming. This has led to quantum speed-ups for the Steiner Tree problem [11], the graph coloring problem [15], and the subset sum problem [1].

The purpose of this work is to adapt the seminal idea of [2] to NP-hard scheduling problems [17] that satisfy the following property: for a given set of jobs J, the optimal solution for J is the

best concatenation of optimal solutions for X and $J \setminus X$ among all $X \subset J$ such that $|X| = |J|/2$ (modulo an additive term that arises in the concatenation). This adaptation requires to introduce a pseudo-polynomial term in the state space of the dynamic programming as well as the aforementioned additive term. We thus obtain an extension of the Dynamic Programming Across the Subsets (DPAS) that many scheduling problems satisfy [17]. Herein, we focus on single-machine scheduling problems and show that our bounded-error hybrid quantum-classical algorithm improves the best-known classical exponential complexities, where in some cases a pseudo-polynomial factor $\sum p_j$ appears. We illustrate it with three examples: minimizing the total weighted completion time with deadlines, minimizing the total weighted completion time with precedence constraints, and minimizing the total weighted tardiness. We summarize in Table 1 the different complexities.

Problem	Quantum-DDPAS	Best classical algorithm
$1\|\tilde{d}_j\|\sum w_jC_j$	$O^*\left(\sum p_j \cdot 1.728^n\right)$	$O^*(2^n)$ (DPAS)
$1\|\|\sum w_jT_j$	$O^*\left(\sum p_j \cdot 1.728^n\right)$	$O^*(2^n)$ (DPAS)
$1\|prec\|\sum w_jC_j$	$O^*\left(1.728^n\right)$	$O^*((2-\epsilon)^n)$, for small ϵ [4]

Table 1: Comparison of complexities between our hybrid algorithm Quantum-DDPAS and the best-known classical algorithm

The rest of the paper is structured as follows. We detail in Section 2 the required property and give examples of single-machine scheduling problems that satisfy it. Then, we describe in Section 3 the hybrid algorithm that solves the problems of interest, recalling basic notions of quantum complexity. Appendix A recalls well-known bounds useful to derive the complexities of the algorithm while Appendix B provides a detailed proof of the correctness of our main algorithm.

2 DYNAMIC PROGRAMMING FOR SCHEDULING

Our problems of interest are scheduling problems where solutions are described by permutations of jobs in $[n] := \{1, \dots, n\}$ for $n \in \mathbb{N}$, and that satisfy a certain property discussed below (see Property 3). This essentially consists of single-machine scheduling problems with constraints.

Let $\mathcal{P}$ be the nominal problem we want to solve. We introduce next a family of problems related to $\mathcal{P}$ that will be instrumental in deriving the dynamic programming recursion. Let T be a set of non-negative integers containing 0. We define the family of problems indexed by $J \subseteq [n]$ and $t \in T$:

$$P(J,t): \quad \min_{\pi \in \Pi(J,t)} f(\pi, J, t), \tag{1}$$

where $\Pi(J,t) \subseteq \mathfrak{S}_J$ is the set of feasible permutations of J according to potential constraints and $f(.,J,t)$ is the objective function. We note $\mathrm{OPT}[J,t]$ the optimal value of $P(J,t)$. With these notations, the nominal problem $\mathcal{P}$ can be cast as follows:

$$\mathcal{P} = P([n], 0).$$

2.1 Dynamic Programming Across the Subsets

We suppose in what follows that $\mathcal{P}$ can be solved by DPAS (Dynamic Programming Across the Subsets). It means that the family of problems must satisfy the following DPAS property.

Property 1 (DPAS). *Let $t_0 \in T$. Problem $P([n], t_0)$ can be solved by DPAS if there exists a function $h : 2^n \times [n] \times T \to \mathbb{R}$, computable in polynomial time, such that the following holds:*

$$\mathrm{OPT}[J, t_0] = \begin{cases} \min\limits_{j \in J} \left\{ \mathrm{OPT}[J \setminus \{j\}, t_0] + h(J, j, t_0) \right\} & \forall J \subseteq [n] \\ \mathrm{OPT}[\emptyset, t_0] = 0 \end{cases} \tag{2}$$

Notice the presence of the additional parameter t_0 in the above definitions, which is typically absent in the scheduling literature. In particular, t_0 is a constant throughout the whole recursion (2) and does not impact the resulting computational complexity. The use of that extra parameter defined in Equation (1) and in Property 1 shall be necessary later when applying our hybrid algorithm.

Lemma 2 (DPAS complexity). *DPAS solves $\mathcal{P}$ in $O^*(2^n)$.*

Proof. We compute Equation (2) for all J such that $|J| = k$, and for $t_0 = 0$, starting from $k = 1$ to $k = n$. For a given J, the values $\{\mathrm{OPT}[J \setminus \{j\}, 0] : j \in J\}$ are known, so $\mathrm{OPT}[J, 0]$ is computed in time $poly(n) \cdot k$ according to Equation (2). Eventually, the total complexity of computing $\mathrm{OPT}[[n], 0]$ is

$$\sum_{k=1}^{n} poly(n) k \binom{n}{k} = poly(n) \cdot n \cdot 2^{n-1} = O^*(2^n).$$

□

In this paper, we consider a family of problems that not only satisfy Property 1, but also the Dichotomic DPAS property below.

Property 3 (Dichotomic DPAS). *Let $t_0 \in T$. Problem $P([n], t_0)$ can be solved by Dichotomic DPAS if there exist three functions $t_1 : 2^n \times 2^n \times T \to T$, $t_2 : 2^n \times 2^n \times T \to T$ and $g : 2^n \times 2^n \times T \to \mathbb{R}$, computable in polynomial time, such that, for all $J \subseteq [n]$ of even cardinality:*

$$\mathrm{OPT}[J, t_0] = \min_{\substack{X \subseteq J \\ |X| = \frac{|J|}{2}}} \left\{ \mathrm{OPT}[X, t_1(J, X, t_0)] + g(J, X, t_0) + \mathrm{OPT}[J \setminus X, t_2(J, X, t_0)] \right\} \tag{3}$$

Notice that if $\mathrm{OPT}[X, t]$ for $X \subseteq [n]$ and $t \in T$ is infeasible, then by convention $\mathrm{OPT}[X, t] = +\infty$. Furthermore, differently from the previous recurrence (2), recurrence (3) now calls $\mathrm{OPT}[X', t']$ for t' that may be different than t_0. This has an impact when deriving the computational complexity of the algorithm in the next lemma.

Lemma 4 (Dichotomic DPAS complexity). *Dichotomic DPAS solves $\mathcal{P}$ in $O^*(|T| \cdot C(n))$ where $C(n) = \omega(2^n)$.*

Proof. We compute Equation (3) for all J such that $|J| = 2^k$, and for all $t \in T$, starting from $k = 1$ to $k = \log_2(n)$. For a given J, the values $\{\mathrm{OPT}[X, t'] : X \subseteq J \text{ s.t. } |X| = |J|/2, t' \in T\}$ are known, so $\mathrm{OPT}[J, t]$ is computed in time $poly(n)\binom{2^k}{2^{k-1}}$ according to Equation (3) (the computation of t_1,t_2 and g is polynomial). Thus, computing all $\mathrm{OPT}[J, t]$ for any J of size 2^k and $t \in T$ is done in

time $|T|poly(n)\binom{2^k}{2^{k-1}}\binom{n}{2^k}$. Eventually, the total complexity is equal to

$$B(n) = |T|poly(n)\sum_{k=1}^{\log_2(n)}\binom{2^k}{2^{k-1}}\binom{n}{2^k}.$$

We show that $\frac{2^n}{B(n)} \to 0$. Let us first consider the sub-sequence $(B(2^i))_{i\in\mathbb{N}}$. For $n = 2^i$, a lower bound of $B(n)$ is the sum of the two last terms: $B(n) > |T|poly(n)(\binom{n}{n/2} + \binom{n}{n/2}\binom{n/2}{n/4}) \approx C|T|poly(n)\frac{2^{1.5n}}{n}$, where C is a constant. Moreover, the sequence $(B(n))_{n\in\mathbb{N}}$ is increasing. Thus, C dominates $n \to 2^n$ asymptotically, namely $C(n) = \omega(2^n)$. □

Notice that solving $\mathcal{P}$ using only Dichotomic DPAS is worse than using only DPAS. However, we describe in the next section a hybrid algorithm we call Quantum Dichotomic DPAS (Q-DDPAS) that improves the complexity of solving $\mathcal{P}$ by combining DPAS and Dichotomic DPAS with Grover search. Before introducing this algorithm, we give some examples of single-machine scheduling problems that satisfy the Dichotomic DPAS property and then can be solved with Q-DDPAS.

2.2 Scheduling Examples

Let us begin with the scheduling problem with deadline constraints and minimization of the total weighted completion time.

Example 5 (Minimizing the total weighted completion time with deadlines). *For each job $j \in [n]$, we are given a weight w_j, a processing time p_j, and a deadline $\tilde{d}_j$. We note $p(J) = \sum_{j\in J} p_j$ and $T = [\![0, p([n])]\!]$. For each $J \subseteq [n]$ and $t \in T$, we consider the problem $P(J,t)$ where*

$$\Pi(J,t) = \{\pi \in \mathfrak{S}_J \mid C_j(\pi) \leq \tilde{d}_j - t, \forall j \in J\},$$

where C_j is the completion time of job j, and for $\pi \in \Pi(J,t)$:

$$f(\pi, J, t) = \sum_{j\in J} w_j(C_j(\pi) + t).$$

$P(J,t)$ represents the problem of finding the best feasible solution for jobs in J supposing that starting time is t, and not 0 as usual. Our problem of interest is $\mathcal{P} = P([n], 0)$, often referred to as $1|\tilde{d}_j|\sum_j w_jC_j$ in the scheduling literature. It can be solved by DPAS. Indeed, Equation (2) is valid with: $\forall J \subseteq [n], \forall j \in J, \forall t \in T$,

$$h(J,j,t) = \begin{cases} w_j(p(J) + t) & \text{if } \tilde{d}_j \geq p(J) + t \\ +\infty & \text{else} \end{cases}$$

where the computation of h is polynomial (linear). This family of problems also satisfies the Dichotomic DPAS property. Indeed, Equation (3) is valid for the following functions:

$\forall X \subseteq J \subseteq [n] s.t. |X| = |J|/2, \forall t \in T$,

$$t_1(J,X,t) = t$$
$$t_2(J,X,t) = t + p(X)$$
$$g(J,X,t) = 0$$

We present another problem that satisfies the Dichotomic DPAS property, which is the scheduling problem with minimization of the total weighted tardiness.

Example 6 (Minimizing the total weighted tardiness). *For each job $j \in [n]$, we are given a weight w_j, a processing time p_j, and a duedate d_j. We note $p(J) = \sum_{j\in J} p_j$. Let $T = [\![0, p([n])]\!]$. For each $J \subseteq [n]$ and $t \in T$, we consider the problem $P(J,t)$ where*

$$\Pi(J,t) = \mathfrak{S}_J,$$

and for $\pi \in \Pi(J,t)$:

$$f(\pi, J, t) = \sum_{j\in J} w_j \max(0, C_j(\pi) - d_j + t),$$

where C_j is the completion time of job j, and $\max(0, C_j - d_j + t)$ represents the tardiness of job j for the effective duedate *$d_j - t$. Our problem of interest is $\mathcal{P} = P([n], 0)$, often referred to as $1||\sum_j w_jT_j$ in the scheduling literature. This problem can be solved by DPAS. Indeed, Equation (2) is valid with: $\forall J \subseteq [n], \forall j \in J, \forall t \in T$,*

$$h(J,j,t) = w_j \max(0, p(J) - d_j + t),$$

where the computation of h is polynomial (linear). This family of problems also satisfies the Dichotomic DPAS. Indeed, Equation (3) is valid for the following functions:

$\forall X \subseteq J \subseteq [n] s.t. |X| = |J|/2, \forall t \in T$,

$$t_1(J,X,t) = t$$
$$t_2(J,X,t) = t + p(X)$$
$$g(J,X,t) = 0$$

We end with the example of the scheduling problem with precedence constraints and minimization of the total weighted completion time.

Example 7 (Minimizing the total weighted completion time with precedence constraints). *We are given, for each job $j \in [n]$ a processing time p_j and a weight w_j, and a set of precedence constraints $E = \{(i,j) : i \prec j\}$ that contains all pairs of jobs (i,j) such that i precedes j. We note $p(J) = \sum_{j\in J} p_j$. Let $T = \{0\}$. Here, the family of problems under consideration is indexed only by the chosen subset of $[n]$. Thus, for each $J \subseteq [n]$, we consider the problem $P(J,0)$ where*

$$\Pi(J,0) = \{\pi \in \mathfrak{S}_J \mid \pi \text{ respects } E\},$$

and for $\pi \in \Pi(J,0)$:

$$f(\pi, J, 0) = \sum_{j\in J} w_jC_j(\pi),$$

where C_j is the completion time of job j. Our problem of interest is $\mathcal{P} = P([n], 0)$, often referred to as $1|prec|\sum_j w_jC_j$ in the scheduling literature. It can be solved by DPAS. Indeed, Equation (2) is valid for:

$$\forall J \subseteq [n], \forall j \in J,\ h(J,j,0) = \begin{cases} +\infty & \text{if } \exists (j,k) \in E | k \in J \\ w_jp(J) & \text{else} \end{cases}$$

where the computation of h is polynomial (quadratic). This family of problems also satisfies the Dichotomic DPAS. Indeed, Equation (3) is valid for the following functions : $\forall X \subseteq J \subseteq [n]$ such that $|X| = |J|/2$,

$$t_1(J,X,0) = 0$$
$$t_2(J,X,0) = 0$$
$$g(J,X,0) = \begin{cases} +\infty & \text{if } \exists (j,k) \in E | j \in J \setminus X \text{ and } k \in X \\ p(X) \cdot \sum_{j\in J\setminus X} w_j & \text{else} \end{cases}$$

where the computation of g is polynomial (quadratic).

3 QUANTUM DICHOTOMIC DPAS ALGORITHM

In this section, we introduce a hybrid bounded-error algorithm called Quantum Dichotomic DPAS (Q-DDPAS) that solves scheduling problems satisfying the Dichotomic DPAS property described in the last section. It is an adaptation of the algorithm in [2]. We describe the quantum part of our algorithm in the gate-based quantum computing model, namely, as a quantum circuit decomposed into single and two-qubit quantum gates. The computational time of such a quantum circuit is quantified by the number of these elementary quantum operations [12]. Henceforth, we assume to have random access to quantum memory (QRAM) [7]. Notice that this is a strong assumption because QRAM is not available on current universal quantum hardware and is not expected to be so in the near future.

3.1 Preliminaries

We begin with some notions of complexity for quantum circuits and some notations for the description of Q-DDPAS.

Definition 8. *Let us consider a family of quantum circuits $(Q_n)_{n\in\mathbb{N}}$ of complexity $O(C(n))$, meaning that Q_n is a circuit that applies on n qubits and contains $f(n)$ universal quantum gates, where $f(n) = O(C(n))$. This family is* efficient *if $C(n) = n^\alpha$ for $\alpha > 0$.*

Observation 9 (Complexity of quantum circuits). *Let U_1 and U_2 be two quantum circuits, with complexity $O(C_1(n))$ and $O(C_2(n))$, respectively. The complexity of the composition $U_1 \cdot U_2$ is*

$$O(C_1(n) + C_2(n)) = O(\max(C_1(n), C_2(n))) .$$

The tensor product $U_1 \otimes U_2$ has the same complexity.

Observation 10 (Classical algorithm into quantum circuit). *Any classical algorithm $\mathcal{A}$ can be described as a quantum circuit $U_\mathcal{A}$. The complexity of $U_\mathcal{A}$ is equal to the complexity of $\mathcal{A}$.*

We define two useful sets for the description of our algorithm, both indexed by a subset and a parameter, (J, t). Essentially, the first set $\Lambda(J, t)$ contains all the possible balanced bi-partitions of J and the associated parameter values of t_1 and t_2. The second set $\Omega(J, t)$ contains the optimal solutions for each bi-partition in $\Lambda(J, t)$.

Definition 11 (Sets Ω and Λ). *For $J \subseteq [n]$ such that $|J|$ is even and for $t \in T$, we define the set*

$$\Lambda(J,t) = \left\{ (X, t_1(J,X,t), J \setminus X, t_2(J,X,t)) : X \subseteq J, |X| = \frac{|J|}{2} \right\},$$

and the set

$$\Omega(J,t) = \left\{ (X, \mathrm{OPT}[X, t_1(J,X,t)], J \setminus X, \mathrm{OPT}[J \setminus X, t_2(J,X,t)], t) : X \subseteq J, |X| = \tfrac{|J|}{2} \right\} .$$

Let us introduce the quantum circuits that constitute the building blocks of our algorithm, and let us provide for each of them their complexity. The two first circuits U_Λ and U_Ω amount to put into uniform superposition the elements of Λ, respectively Ω.

Definition 12 (Circuit U_Λ). *For $J \subseteq [n]$ such that $|J|$ is even, and for $t \in T$, we define U_Λ as follows:*

$$U_\Lambda |J\rangle |t\rangle |0\rangle^{\otimes 6} = |J\rangle |t\rangle \sum_{(\lambda_1^s, \lambda_1^t, \lambda_2^s, \lambda_2^t) \in \Lambda(J,t)} \frac{1}{\sqrt{|\Lambda(J,t)|}} \left|\lambda_1^s\right\rangle \left|\lambda_1^t\right\rangle |0\rangle \left|\lambda_2^s\right\rangle \left|\lambda_2^t\right\rangle |0\rangle .$$

Notice that we index the objects that represent sets by s, and the objects that represent parameters in T by t.

Property 13 (Complexity of U_Λ). *The complexity of U_Λ is polynomial in the size of the input.*

Proof. First, let us prove that the construction of the superposition of subsets of J of size $|J|/2$ is polynomial. Let $J \subseteq [n]$ of size m (we suppose m to be even). Let us prove that the construction of a quantum superposition of balanced bi-partition (both subsets are of size $\frac{m}{2}$) of J can be done in polynomial time.

We know that there are $\binom{m}{m/2}$ balanced bi-partitions. It is possible to construct implicitly a bijection

$$\sigma : \left[\!\left[1, \ldots, \binom{m}{m/2} \right]\!\right] \to \left\{ (X, J \setminus X) : |X| = \frac{m}{2} \right\}$$

that enumerates the balanced bi-partitions (it requires making a bijection between J and $[\![1, \ldots, m]\!]$). Notice that the construction of this bijection does not depend on the values of J. For a given i in $\left[\!\left[1, \ldots, \binom{m}{m/2} \right]\!\right]$, the computation of $\sigma(i) = (X_i, J \setminus X_i)$ is polynomial.

Thus,

$$|i\rangle |0\rangle \to |i\rangle |X_i\rangle |J \setminus X_i\rangle$$

is done with a quantum circuit with a polynomial complexity, for any J. It implies that

$$\sum_{i=1}^{\binom{m}{m/2}} |i\rangle |0\rangle \to \sum_{i=1}^{\binom{m}{m/2}} |i\rangle |X_i\rangle |J \setminus X_i\rangle$$

is also done in polynomial time, resulting from the application of Hadamard gates.

Eventually, the computation of functions t_1 and t_2 is polynomial (see hypothesis in Dichotomic DPAS Property 3). Thus, the complexity of U_Λ is polynomial. □

Definition 14 (Circuit U_Ω). *For $J \subseteq [n]$ such that $|J|$ is even, and for $t \in T$, we define U_Ω as follows:*

$$U_\Omega |J\rangle |t\rangle |0\rangle = |J\rangle |t\rangle \sum_{\omega \in \Omega(J,t)} \frac{1}{\sqrt{|\Omega(J,t)|}} |\omega\rangle .$$

Property 15 (Complexity of U_Ω). *Let J be the input set. If we suppose to have stored in the QRAM the values $\mathrm{OPT}[X, t]$ for all $X \subseteq J$ such that $|X| = n/4$ and for all $t \in T$, the complexity of U_Ω is polynomial in the size of the input.*

Proof. The proof follows essentially the same lines as the proof of Property 13. The quantum superposition of subsets is done in polynomial time, and instead of computing t_1 and t_2, we get values in the QRAM in constant time. □

Definition 16 (Circuit U_r). *Let $r : \Omega(J,t) \to \mathbb{Z}$ be the function:*

$$r(\omega_1^s, \omega_1^v, \omega_2^s, \omega_2^v, \omega^t) = \omega_1^v + \omega_2^v + g(\omega_1^s \cup \omega_2^s, \omega_1^s, \omega^t)\,.$$

We note U_r the quantum circuit corresponding to r, namely:

$$\forall(\omega_1^s, \omega_1^v, \omega_2^s, \omega_2^v, \omega^t) \in \Omega(J,t),\ U_r\,|\omega\rangle\,|0\rangle = |\omega\rangle\,|r(\omega)\rangle\,,$$

where $|\omega\rangle = \left|\omega_1^s\right\rangle \left|\omega_1^v\right\rangle \left|\omega_2^s\right\rangle \left|\omega_2^v\right\rangle \left|\omega^t\right\rangle$ is encoded in five registers. Notice that we index the objects that represent numerical values by v.

Property 17 (Complexity of U_r). *The complexity of U_r is polynomial in the size of the input.*

Proof. The computation of g is polynomial (see hypothesis in Dichotomic DPAS Property 3). It implies that the computation of r is polynomial, and thus that U_r has a polynomial complexity (Property 10). □

Definition 18 (Circuit U_{QMF}). *Let $f : [n] \to \mathbb{Z}$ be a function and let U_f be its corresponding quantum circuit, specifically,*

$$U_f\,|i\rangle\,|0\rangle = |i\rangle\,|f(i)\rangle\,,\ \forall i \in [n]\,.$$

We note $U_{\text{QMF}}[U_f]$ the quantum circuit corresponding to the Quantum Minimum Finding algorithm [5] that computes with high probability the minimum value of f and the corresponding minimizer:

$$U_{\text{QMF}}[U_f] \sum_{i=1}^{n} \frac{1}{\sqrt{n}}\,|i\rangle\,|0\rangle\,|0\rangle = \sum_{i=1}^{n} \frac{1}{\sqrt{n}}\,|i\rangle \left|\arg\min_{i\in[n]}\{f(i)\}\right\rangle \left|\min_{i\in[n]}\{f(i)\}\right\rangle\,.$$

Property 19 (Complexity of U_{QMF} [5]). *The complexity of the Quantum Minimum Finding algorithm is $O\left(\sqrt{n}\cdot C_f(n)\right)$, where n is the size of the domain of f and $O(C_f(n))$ is the complexity of the circuit U_f. Thus, the complexity of $U_{\text{QMF}}[U_f]$ is*

$$O\left(\sqrt{n}\cdot C_f(n)\right)$$

according to Observation 10.

In the next section, we use some notations as follows. Let $reg = |q_1\rangle \ldots |q_n\rangle$ be a register of n qubits and U be an operator acting on k qubits, with $k < n$. Let I be a k-tuple of distinct indices in $[n]$, $I = (i_1, \ldots, i_k)$. We denote by U^I the operator acting on the full register reg, that applies U on $\left|q_{i_1}\right\rangle \ldots \left|q_{i_k}\right\rangle$, and applies Id on the remaining qubits. For instance, if I is the tuple of contiguous indices $(3, \ldots, k+3)$ with $k < n-3$, then

$$U^I := Id^{\otimes 2} \otimes U \otimes Id^{\otimes n-k-3}\,.$$

For $I = (i_1, \ldots, i_k)$ and $J = (j_1, \ldots, j_l)$ two distinct tuples in $[n]$ (k-tuple and l-tuple where $i \neq j, \forall (i,j) \in I \times J$) , we note $I \oplus J$ the concatenation of I and J, namely $I \oplus J = (i_1, \ldots, i_k, j_1, \ldots, j_l)$.

Let us denote the indexes related to the quantum circuit U_f as

$$U_f\ \underbrace{|i\rangle}_{I}\ \underbrace{|0\rangle}_{J} = \underbrace{|i\rangle}_{I}\ \underbrace{|f(i)\rangle}_{J}\,.$$

To clarify the computations detailed next, we index the corresponding QMF operator as $U_{\text{QMF}}[U_f^I]$. We omit the index J because this is an *auxiliary* register that does not appear in the output of $U_{\text{QMF}}[U_f]$.

3.2 Description of the Algorithm

We describe the Quantum Dichotomic DPAS (Q-DDPAS) algorithm as an adaptation of [2] for scheduling problems satisfying the Dichotomic DPAS property. Without loss of generality, we assume that 4 divides n. The hybrid quantum-classical algorithm Q-DDPAS consists of two steps:

(1) Classical part: For each X of size $n/4$ and all $t \in T$, solve by classical DPAS the problem $P(X,t)$. Store the results in the QRAM as tuples $(X, t, \text{OPT}[X,t], \pi^*[X,t])$, where $\pi^*[X,t]$ is the optimal permutation corresponding to $\text{OPT}[X,t]$.
(2) Quantum part:
   (a) Apply quantum circuit

$$U_{QDDPAS} = U_{\text{recur}} U_{\text{ini}}$$

   to the initial state

$$|\text{ini}\rangle = \underbrace{|[n]\rangle}_{I^1}\ \underbrace{|0\rangle}_{I^2}\ \underbrace{|0\rangle^{\otimes 3}}_{I^3}\ \underbrace{|0\rangle^{\otimes 3}}_{I^4}\ \underbrace{|0\rangle^{\otimes 3}}_{I^5}\ \underbrace{|0\rangle^{\otimes 3}}_{I^6},$$

   where the tuples indexing the different registers are decomposed as follows:

$$\begin{aligned} I^1 &= I_1^1 \oplus I_2^1 \\ I^2 &= I_1^2 \oplus I_2^2 \oplus I_3^2 \\ I^3 &= I_1^3 \oplus I_2^3 \\ I^4 &= I_1^4 \oplus I_2^4 \oplus I_3^4 \\ I^5 &= I_1^5 \oplus I_2^5 \\ I^6 &= I_1^6 \oplus I_2^6 \end{aligned}$$

   and where

$$U_{\text{ini}} = (U_\Omega^{I^2} \otimes U_\Omega^{I^4}) \cdot U_\Lambda^{I^1 \oplus I^2 \oplus I^4}\,, \tag{4}$$

   and

$$U_{\text{recur}} = U_{\text{QMF}}^{I_1^2 \oplus I_3^3 \oplus I_1^4 \oplus I_3^5 \oplus I_2^1 \oplus I^6}[U_{\text{recur1}}]\,, \tag{5}$$

   where

$$U_{\text{recur1}} = U_r^{I_1^2 \oplus I_2^3 \oplus I_1^4 \oplus I_2^5 \oplus I_2^1} \cdot \left(U_{\text{QMF}}^{I_2^2 \oplus I^3}[U_r^{I_3^2}] \otimes U_{\text{QMF}}^{I_3^4 \oplus I^5}[U_r^{I_3^4}]\right)\,.$$

   (b) Measure register of indexes I_2^6 to find the optimal value $\text{OPT}[[n], 0]$.

The main idea of this algorithm is as follows. First, we compute classically by DPAS the optimal values of all subproblems scheduling with $\frac{n}{4}$ jobs. Second, we call recursively two times QMF to find optimal values of subproblems scheduling with $n/2$ jobs and eventually with n jobs (corresponding to the initial problem).

Theorem 20. *The bounded-error Q-DDPAS algorithm solves $\mathcal{P}$ in $O^*(|T| \cdot 1.754^n)$.*

The proof of Theorem 20 relies on the two lemmas introduced next. However, before stating and proving these lemmas, we observe that the complexity of Q-DDPAS can be further reduced by performing a third call to Dichotomic DPAS recurrence (3) as suggested in [2].

Observation 21. *A slight modification of Q-DDPAS reduces the complexity to $O^*(|T| \cdot 1.728^n)$.*

For the sake of clarity, we will prove Observation 21 only after having proved Theorem 20. We now introduce the two lemmas necessary to prove Theorem 20.

Lemma 22. *The optimal value of $\mathcal{P}$ is stored in the register of indexes I_2^6 by Q-DDPAS with high probability.*

Proof. We provide next a sketch of the proof, referring to Appendix B for the details of the computations. The main idea is to compute the first terms by classical DPAS, and then apply recursively twice Equation (3), which is solved by QMF:

- *Classical part:* Compute by classical DPAS the values $\mathrm{OPT}[X,t]$ for all X of size $n/4$ and for all $t \in T$. Store the results in the QRAM.
- *Quantum part:*
  - For each J of size $n/2$ and $t \in T$, compute $\mathrm{OPT}[J,t]$ through Equation (3) combined with QMF.
  - Compute $\mathrm{OPT}[[n],0]$ with Equation (3) combined with QMF.

We now give some intuition on the effect of the quantum circuit $U_{QDDPAS} = U_{\text{recur}}U_{\text{ini}}$ and start by explaining the effect of U_{ini} defined in (4). First, the application of U_Λ superposes all elements of $\Lambda([n],0)$ in the registers of indexes I^2 (partition J) and I^4 (partition $[n] \setminus J$). This essentially amounts to superpose all the $\binom{n}{n/2}$ bi-partitions of $[n]$ where each partition is of size $n/2$ (parameters t included). Then, we apply U_Ω on register of index I^2, resp. I^4. This superposes all elements of $\Omega(J,t)$ (for a J of size $n/2$ and $t \in T$ previously described in registers of indexes I^2, resp. I^4). This essentially amounts to superpose all the $\binom{n/2}{n/4}$ bi-partitions of $[n]$ where each partition is of size $n/2$, parameters t included, and the optimal value associated already stored in the QRAM.

Let us explain the effect of U_{recur} defined in (5). The application of $U_{\text{QMF}}[U_r]$ on a register encoding $(J,\ t)$ and the superposition of elements of $\Omega(J,t)$ stores (with high probability) in an output register $\mathrm{OPT}[J,t]$ according to the Dichotomic DPAS Property 3. Thus, $U_{\text{QMF}}[U_r]$ on register of index I^2, resp. I^4, superposes all $\mathrm{OPT}[J,t]$ in I^3, resp. I^5. In other words, the circuit $U_{\text{QMF}}^{I_3^2 \oplus I^3}[U_r^{I_3^2 \oplus I_1^3}] \otimes U_{\text{QMF}}^{I_3^4 \oplus I^5}[U_r^{I_3^4 \oplus I_1^5}]$ that appears in U_{recur1} superposes (with high probability) all optimal values of Equation (3) for J of size $n/2$. Now that the optimal values are known for sets of size $n/2$ (before, we only knew optimal values for sets of size $n/4$), we apply one more time $U_{\text{QMF}}[U_r]$ on these new registers: it outputs $\mathrm{OPT}[[n],0]$ on the register of index I_2^6 with high probability. □

Lemma 23. *The complexity of Q-DDPAS is $O^*(|T| \cdot 1.754^n)$.*

Proof. Let us compute the complexity of this algorithm. First, we compute the complexity of the classical part. The proof of Lemma 2 shows that solving all $\mathrm{OPT}[X,t]$ for all X of size $n/4$ and for all $t \in T$ is done by DPAS in time

$$|T| poly(n) \sum_{k=1}^{n/4} k\binom{n}{k} = O^*\left(|T|\binom{n}{\leq n/4}\right).$$

Thus, because $O^*\left(\binom{n}{\leq n/4}\right) = O^*(2^{0.811n})$ (see Equation (6)), the complexity of the classical part is

$$O^*(|T| \cdot 2^{0.811n}).$$

Second, let us compute the complexity of the quantum part (using Property 10).

- The complexity of U_{ini} is polynomial in n. Indeed, U_Λ is polynomial in n (Property 13). Moreover, U_Ω is also polynomial in n: the classical part stored in the QRAM all $\mathrm{OPT}[X,t]$ for X of size $n/4$ and $t \in T$ (Property 15).
- The complexity of U_{recur} is $O^*\left(\sqrt{\binom{n}{n/2}\binom{n/2}{n/4}}\right)$. Indeed, both terms $U_{\text{QMF}}[U_r]$ in U_{recur1} have a polynomial complexity for U_r and find the minimum of functions with a domain of size $\binom{n/2}{n/4}$. Thus, each complexity of these two factors is $O^*\left(\sqrt{\binom{n/2}{n/4}}\right)$, and so is the complexity of the tensor product. The circuit U_{recur1} has the same complexity because of the composition with U_r that is polynomial. The circuit U_{recur} finds the minimum of a function with a domain of size $\binom{n}{n/2}$ described by the corresponding quantum circuit U_{recur1} above. Thus, its complexity is $O^*\left(\sqrt{\binom{n}{n/2}\binom{n/2}{n/4}}\right)$.

Because $O^*\left(\sqrt{\binom{n}{n/2}\binom{n/2}{n/4}}\right) = O^*(2^{0.75n})$ (see Equation (8)), the complexity of the quantum part is

$$O^*(2^{0.75n}).$$

Eventually, the complexity of Q-DDPAS is

$$O^*\left(2^{0.75n} + |T| \cdot 2^{0.811n}\right) = O^*\left(|T| \cdot 2^{0.811n}\right) = O^*\left(|T| \cdot 1.754^n\right).$$

□

Proof of Theorem 20. Follows directly from Lemmas 22 and 23. □

Proof of Observation 21. The slight modification of Q-DDPAS amounts to add a level of recurrence in the quantum part, but instead of searching for the best concatenation among all the bi-partition of size $(n/8, n/8)$ (i.e. solving Equation (3) for $|J| = n/4$), we search for the best concatenation among all the bi-partition of size $(0.945 \cdot \frac{n}{4}, 0.055 \cdot \frac{n}{4})$, i.e. solving

$$\mathrm{OPT}[J,t] = \min_{\substack{X \subseteq J \\ |X| = 0.945|J|}} \left\{\mathrm{OPT}[X, t_1(J,X,t)] + g(J,X,t) + \mathrm{OPT}[J \setminus X, t_2(J,X,t)]\right\}$$

This further recurrence implies that:

- the classical part computes $\mathrm{OPT}[J,t]$ for J of size $0.945 \cdot \frac{n}{4}$ and $0.055 \cdot \frac{n}{4}$. Its complexity is then $O^*\left(|T|\binom{n}{\leq 0.945 \cdot \frac{n}{4}}\right) = O^*(|T| \cdot 2^{0.789n})$ (see Equation (7)).
- the quantum part applies three levels of recurrence of QMF, finding the minimum over functions with a domain of size $\binom{n}{n/2}$, $\binom{n/2}{n/4}$ and $\binom{n/4}{0.945 \cdot n/4}$ respectively. Its complexity is

then $O^*\left(\sqrt{\binom{n}{n/2}\binom{n/2}{n/4}\binom{n/4}{0.945\cdot n/4}}\right) = O^*(2^{0.789n})$ (see Equation (9)).

The quantum part and the classical part have the same complexity, thus the total complexity of Q-DDPAS is the same, namely $O^*(2^{0.789n}) = O^*(1.728^n)$. □

We summarized in Table 1 the complexities of solving the scheduling problems studied in Section 2 with Q-DDPAS and compare them with the complexities of the best-known current classical algorithms. Q-DDPAS improves the complexity of the exponent but sometimes at the cost of a pseudo-polynomial factor ($\sum p_j$ for problems $1|\tilde{d}_j|\sum_j w_jC_j$ and $1||\sum_j w_jT_j$).

4 CONCLUSION

This paper extends the hybrid algorithm of [2] to scheduling problems that satisfy the Dichotomic DPAS property. Such problems, which are often solved in $O^*(2^n)$ by classical DPAS, are solved by our bounded-error algorithm in $O^*(|T|\cdot 1.754^n)$, where $|T|$ is meant to be at most pseudo-polynomial in the size of the problem. We detail the application of the resulting hybrid algorithm on three single-machine scheduling problems ($1|\tilde{d}_j|\sum_j w_jC_j$, $1||\sum_j w_jT_j$ and $1|prec|\sum_j w_jC_j$), showing a reduction of the exponent compared to the best-known classical complexity. We notice that a pseudo-polynomial factor appears in the complexity of two out of three problems. Future works will seek to extend these results to other scheduling problems for which the Dichotomic DPAS holds, such as the 3-machine flowshop scheduling problem [14].

REFERENCES

[1] Jonathan Allcock, Yassine Hamoudi, Antoine Joux, Felix Klingelhöfer, and Miklos Santha. 2022. Classical and quantum algorithms for variants of subset-sum via dynamic programming. In *30th Annual European Symposium on Algorithms (ESA 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

[2] Andris Ambainis, Kaspars Balodis, Jānis Iraids, Martins Kokainis, Krišjānis Prūsis, and Jevgēnijs Vihrovs. 2019. Quantum speedups for exponential-time dynamic programming algorithms. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 1783–1793.

[3] Marco Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, et al. 2021. Variational quantum algorithms. *Nature Reviews Physics* 3, 9 (2021), 625–644.

[4] Marek Cygan, Marcin Pilipczuk, Michał Pilipczuk, and Jakub Onufry Wojtaszczyk. 2014. Scheduling partially ordered jobs faster than 2^n. *Algorithmica* 68 (2014), 692–714.

[5] Christoph Durr and Peter Hoyer. 1996. A quantum algorithm for finding the minimum. *arXiv preprint quant-ph/9607014* (1996).

[6] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. 2014. A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028* (2014).

[7] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. 2008. Quantum random access memory. *Physical review letters* 100, 16 (2008), 160501.

[8] Camille Grange, Michael Poss, and Eric Bourreau. 2022. An introduction to variational quantum algorithms on gate-based quantum computing for combinatorial optimization problems. *arXiv preprint arXiv:2212.11734* (2022).

[9] Lov K Grover. 1996. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*. 212–219.

[10] Michael Held and Richard M Karp. 1970. The traveling-salesman problem and minimum spanning trees. *Operations Research* 18, 6 (1970), 1138–1162.

[11] Masayuki Miyamoto, Masakazu Iwamura, Koichi Kise, and François Le Gall. 2020. Quantum speedup for the minimum steiner tree problem. In *Computing and Combinatorics: 26th International Conference, COCOON 2020, Atlanta, GA, USA, August 29–31, 2020, Proceedings*. Springer, 234–245.

[12] Ashley Montanaro. 2016. Quantum algorithms: an overview. *npj Quantum Information* 2, 1 (2016), 1–8.

[13] Giacomo Nannicini. 2022. Fast quantum subroutines for the simplex method. *Operations Research* (2022). In press.

[14] Lei Shang, Christophe Lenté, Mathieu Liedloff, and Vincent T'Kindt. 2018. Exact exponential algorithms for 3-machine flowshop scheduling problems. *Journal of Scheduling* 21 (2018), 227–233.

[15] Kazuya Shimizu and Ryuhei Mori. 2022. Exponential-time quantum algorithms for graph coloring problems. *Algorithmica* (2022), 1–19.

[16] David Sutter, Giacomo Nannicini, Tobias Sutter, and Stefan Woerner. 2020. Quantum speedups for convex dynamic programming. *arXiv preprint arXiv:2011.11654* (2020).

[17] Vincent T'kindt, Federico Della Croce, and Mathieu Liedloff. 2022. Moderate exponential-time algorithms for scheduling problems. *4OR* (2022), 1–34.

A NOTATIONS AND UPPER BOUNDS

In what follows, we use the notation

$$\binom{n}{\leq k} = \sum_{i=1}^{k}\binom{n}{i}.$$

We also note the binary entropy of $\epsilon \in]0,1[$ the quantity $H(\epsilon) = -(\epsilon\log_2(\epsilon)+(1-\epsilon)\log_2(1-\epsilon))$. We give some useful upper bounds of binomial coefficients [2]:

$$\forall k \in [\![1,n]\!],\ \binom{n}{k} \leq 2^{H\left(\frac{k}{n}\right)\cdot n},$$

$$\forall k \in \left[\!\left[1,\frac{n}{2}\right]\!\right],\ \binom{n}{\leq k} \leq 2^{H\left(\frac{k}{n}\right)\cdot n}.$$

Observe that $\binom{n}{\leq n/4}$ is bounded above by $2^{H\left(\frac{n/4}{n}\right)\cdot n}$, where $H\left(\frac{n/4}{n}\right) = H\left(\frac{1}{4}\right) = -\left(\frac{1}{4}\log_2\left(\frac{1}{4}\right)+\frac{3}{4}\log_2\left(\frac{3}{4}\right)\right) = 0.811$, so we obtain

$$\binom{n}{\leq n/4} \leq 2^{0.811n}. \tag{6}$$

In the same way, we can show that

$$\binom{n}{\leq 0.945\cdot n/4} \leq 2^{0.789n}. \tag{7}$$

Similarly, $\sqrt{\binom{n}{n/2}\binom{n/2}{n/4}}$ is bounded above by $\sqrt{2^{H\left(\frac{n/4}{n/2}\right)\cdot\frac{n}{2}}2^{H\left(\frac{n/2}{n}\right)\cdot n}} = 2^{\frac{1}{2}\left(\frac{1}{2}H\left(\frac{1}{2}\right)+H\left(\frac{1}{2}\right)\right)\cdot n} = 2^{\frac{3}{4}H\left(\frac{1}{2}\right)n}$, where $H\left(\frac{1}{2}\right) = 1$, leading to

$$\sqrt{\binom{n}{n/2}\binom{n/2}{n/4}} \leq 2^{0.75n}. \tag{8}$$

In the same way, we can show that

$$\sqrt{\binom{n}{n/2}\binom{n/2}{n/4}\binom{n/4}{0.945\cdot n/4}} \leq 2^{0.789n}. \tag{9}$$

B DETAILED PROOF OF LEMMA 22

Next, we compute $U_{\text{recur}}U_{\text{ini}}|\text{ini}\rangle$ and show that $\text{OPT}[[n],0]$ is stored in register of indexes I_2^6. First, we compute $U_{\text{ini}}|\text{ini}\rangle$.

$$\begin{aligned}
U_\Lambda^{I^1\oplus I^2\oplus I^4}|\text{ini}\rangle =& U_\Lambda^{I^1\oplus I^2\oplus I^4}\underbrace{|[n]\rangle}_{I^1}\underbrace{|0\rangle}_{I^2}\underbrace{|0\rangle^{\otimes 3}}_{I^3}\underbrace{|0\rangle^{\otimes 2}}_{I^4}\underbrace{|0\rangle^{\otimes 3}}_{I^5}\underbrace{|0\rangle^{\otimes 2}}_{I^6}\underbrace{|0\rangle^{\otimes 2}}_{I^6}\\
=& \underbrace{|[n]\rangle}_{I^1}\underbrace{|0\rangle}_{I^2}\sum_{(\lambda_1^s,\lambda_1^t,\lambda_2^s,\lambda_2^t)\in\Lambda([n],0)}\frac{1}{\sqrt{|\Lambda([n],0)|}}\\
&\underbrace{|\lambda_1^s\rangle|\lambda_1^t\rangle|0\rangle}_{I^2}\underbrace{|0\rangle^{\otimes 2}}_{I^3}\underbrace{|\lambda_2^s\rangle|\lambda_2^t\rangle|0\rangle}_{I^4}\underbrace{|0\rangle^{\otimes 2}}_{I^5}\underbrace{|0\rangle^{\otimes 2}}_{I^6}.
\end{aligned}$$

Thus,

$$
\begin{aligned}
U_{\text{ini}}\left|\text{ini}\right\rangle &= \left(U_\Omega^{I^2}\otimes U_\Omega^{I^4}\right)\cdot U_\Lambda^{I^1\oplus I^2\oplus I^4}\left|\text{ini}\right\rangle \\
&= \left((U_\Omega^{I^2}\otimes U_\Omega^{I^4}\right)\underbrace{\left|[n]\right\rangle\left|0\right\rangle}_{I^1}\sum_{(\lambda_1^s,\lambda_1^t,\lambda_2^s,\lambda_2^t)\in\Lambda([n],0)}\frac{1}{\sqrt{|\Lambda([n],0)|}} \\
&\quad \underbrace{\left|\lambda_1^s\right\rangle\left|\lambda_1^t\right\rangle\left|0\right\rangle}_{I^2}\underbrace{\left|0\right\rangle^{\otimes 2}}_{I^3}\underbrace{\left|\lambda_2^s\right\rangle\left|\lambda_2^t\right\rangle\left|0\right\rangle}_{I^4}\underbrace{\left|0\right\rangle^{\otimes 2}}_{I^5}\underbrace{\left|0\right\rangle^{\otimes 2}}_{I^6} \\
&= \underbrace{\left|[n]\right\rangle\left|0\right\rangle}_{I^1}\sum_{(\lambda_1^s,\lambda_1^t,\lambda_2^s,\lambda_2^t)\in\Lambda([n],0)}\frac{1}{\sqrt{|\Lambda([n],0)|}} \\
&\quad \underbrace{\left|\lambda_1^s\right\rangle}_{I_1^2}\underbrace{\left|\lambda_1^t\right\rangle}_{I_2^2}\left(\sum_{\omega\in\Omega(\lambda_1^s,\lambda_1^t)}\frac{1}{\sqrt{|\Omega(\lambda_1^s,\lambda_1^t)|}}\underbrace{\left|\omega\right\rangle}_{I_3^2}\underbrace{\left|0\right\rangle^{\otimes 2}}_{I^3}\right) \\
&\quad \underbrace{\left|\lambda_2^s\right\rangle}_{I_1^4}\underbrace{\left|\lambda_2^t\right\rangle}_{I_2^4}\left(\sum_{\omega\in\Omega(\lambda_2^s,\lambda_2^t)}\frac{1}{\sqrt{|\Omega(\lambda_2^s,\lambda_2^t)|}}\underbrace{\left|\omega\right\rangle}_{I_3^4}\underbrace{\left|0\right\rangle^{\otimes 2}}_{I^5}\right)\underbrace{\left|0\right\rangle^{\otimes 2}}_{I^6}.
\end{aligned}
$$

Second, we apply the tensor product of the two first QMF to the previous state.

$$
\begin{aligned}
&\left(U_{\text{QMF}}^{I_3^2\oplus I^3}[U_r^{I_3^2}]\otimes U_{\text{QMF}}^{I_3^4\oplus I^5}[U_r^{I_3^4}]\right)\underbrace{\left|[n]\right\rangle\left|0\right\rangle}_{I^1} \\
&\sum_{(\lambda_1^s,\lambda_1^t,\lambda_2^s,\lambda_2^t)\in\Lambda([n],0)}\frac{1}{\sqrt{|\Lambda([n],0)|}} \\
&\underbrace{\left|\lambda_1^s\right\rangle}_{I_1^2}\underbrace{\left|\lambda_1^t\right\rangle}_{I_2^2}\left(\sum_{\omega\in\Omega(\lambda_1^s,\lambda_1^t)}\frac{1}{\sqrt{|\Omega(\lambda_1^s,\lambda_1^t)|}}\underbrace{\left|\omega\right\rangle}_{I_3^2}\underbrace{\left|0\right\rangle^{\otimes 2}}_{I^3}\right) \\
&\underbrace{\left|\lambda_2^s\right\rangle}_{I_1^4}\underbrace{\left|\lambda_2^t\right\rangle}_{I_2^4}\left(\sum_{\omega\in\Omega(\lambda_2^s,\lambda_2^t)}\frac{1}{\sqrt{|\Omega(\lambda_2^s,\lambda_2^t)|}}\underbrace{\left|\omega\right\rangle}_{I_3^4}\underbrace{\left|0\right\rangle^{\otimes 2}}_{I^5}\right)\underbrace{\left|0\right\rangle^{\otimes 2}}_{I^6} \\
&= \underbrace{\left|[n]\right\rangle\left|0\right\rangle}_{I^1}\sum_{(\lambda_1^s,\lambda_1^t,\lambda_2^s,\lambda_2^t)\in\Lambda([n],0)}\frac{1}{\sqrt{|\Lambda([n],0)|}}\underbrace{\left|\lambda_1^s\right\rangle}_{I_1^2}\underbrace{\left|\lambda_1^t\right\rangle}_{I_2^2} \\
&\quad \sum_{\omega\in\Omega(\lambda_1^s,\lambda_1^t)}\frac{1}{\sqrt{|\Omega(\lambda_1^s,\lambda_1^t)|}}\underbrace{\left|\omega\right\rangle}_{I_3^2}\underbrace{\left|\arg\min_\omega r(\omega)\right\rangle\left|\min_\omega r(\omega)\right\rangle}_{I_1^3\otimes I_2^3} \\
&\quad \underbrace{\left|\lambda_2^s\right\rangle}_{I_1^4}\underbrace{\left|\lambda_2^t\right\rangle}_{I_2^4}\sum_{\omega\in\Omega(\lambda_2^s,\lambda_2^t)}\frac{1}{\sqrt{|\Omega(\lambda_2^s,\lambda_2^t)|}}\underbrace{\left|\omega\right\rangle}_{I_3^4} \\
&\quad \underbrace{\left|\arg\min_\omega r(\omega)\right\rangle\left|\min_\omega r(\omega)\right\rangle}_{I_1^5\otimes I_2^5}\underbrace{\left|0\right\rangle^{\otimes 2}}_{I^6}.
\end{aligned}
$$

Thus, we apply the second circuit of QMF.

$$
\begin{aligned}
U_{\text{recur}}U_{\text{ini}}\left|\text{ini}\right\rangle &= U_{\text{QMF}}^{I_1^2\oplus I_3^3\oplus I_1^4\oplus I_3^5\oplus I_2^1\oplus I^6}[U_{\text{recur1}}]U_{\text{ini}}\left|\text{ini}\right\rangle \\
&= U_{\text{QMF}}^{I_1^2\oplus I_3^3\oplus I_1^4\oplus I_3^5\oplus I_2^1\oplus I^6}[U_r^{I_1^2\oplus I_2^3\oplus I_1^4\oplus I_2^5\oplus I_2^1}]\underbrace{\left|[n]\right\rangle\left|0\right\rangle}_{I^1} \\
&\quad \sum_{(\lambda_1^s,\lambda_1^t,\lambda_2^s,\lambda_2^t)\in\Lambda([n],0)}\frac{1}{\sqrt{|\Lambda([n],0)|}}\underbrace{\left|\lambda_1^s\right\rangle}_{I_1^2}\underbrace{\left|\lambda_1^t\right\rangle}_{I_2^2} \\
&\quad \sum_{\omega\in\Omega(\lambda_1^s,\lambda_1^t)}\frac{1}{\sqrt{|\Omega(\lambda_1^s,\lambda_1^t)|}}\underbrace{\left|\omega\right\rangle}_{I_3^2}\underbrace{\left|\arg\min_\omega r(\omega)\right\rangle\left|\min_\omega r(\omega)\right\rangle}_{I_1^3\otimes I_2^3} \\
&\quad \underbrace{\left|\lambda_2^s\right\rangle}_{I_1^4}\underbrace{\left|\lambda_2^t\right\rangle}_{I_2^4}\sum_{\omega\in\Omega(\lambda_2^s,\lambda_2^t)}\frac{1}{\sqrt{|\Omega(\lambda_2^s,\lambda_2^t)|}}\underbrace{\left|\omega\right\rangle}_{I_3^4} \\
&\quad \underbrace{\left|\arg\min_\omega r(\omega)\right\rangle\left|\min_\omega r(\omega)\right\rangle}_{I_1^5\otimes I_2^5}\underbrace{\left|0\right\rangle^{\otimes 2}}_{I^6} \\
&= \underbrace{\left|[n]\right\rangle\left|0\right\rangle}_{I^1}\sum_{(\lambda_1^s,\lambda_1^t,\lambda_2^s,\lambda_2^t)\in\Lambda([n],0)}\frac{1}{\sqrt{|\Lambda([n],0)|}}\underbrace{\left|\lambda_1^s\right\rangle}_{I_1^2}\underbrace{\left|\lambda_1^t\right\rangle}_{I_2^2} \\
&\quad \sum_{\omega\in\Omega(\lambda_1^s,\lambda_1^t)}\frac{1}{\sqrt{|\Omega(\lambda_1^s,\lambda_1^t)|}}\underbrace{\left|\omega\right\rangle}_{I_3^2}\underbrace{\left|\arg\min_\omega r(\omega)\right\rangle\left|\min_\omega r(\omega)\right\rangle}_{I_1^3\otimes I_2^3} \\
&\quad \underbrace{\left|\lambda_2^s\right\rangle}_{I_1^4}\underbrace{\left|\lambda_2^t\right\rangle}_{I_2^4}\sum_{\omega\in\Omega(\lambda_2^s,\lambda_2^t)}\frac{1}{\sqrt{|\Omega(\lambda_2^s,\lambda_2^t)|}}\underbrace{\left|\omega\right\rangle}_{I_3^4} \\
&\quad \underbrace{\left|\arg\min_\omega r(\omega)\right\rangle\left|\min_\omega r(\omega)\right\rangle}_{I_1^5\otimes I_2^5} \\
&\quad \underbrace{\left|\arg\min_{\lambda\in\Lambda([n],0)} r(\lambda_1^s,\min_{\omega\in\Omega(\lambda_1^s,\lambda_1^t)} r(\omega),\lambda_2^s,\min_{\omega\in\Omega(\lambda_2^s,\lambda_2^t)} r(\omega),0)\right\rangle}_{I_1^6} \\
&\quad \underbrace{\left|\min_{\lambda\in\Lambda([n],0)} r(\lambda_1^s,\min_{\omega\in\Omega(\lambda_1^s,\lambda_1^t)} r(\omega),\lambda_2^s,\min_{\omega\in\Omega(\lambda_2^s,\lambda_2^t)} r(\omega),0)\right\rangle}_{I_2^6}.
\end{aligned}
$$

According to definition of r and the Dichotomic DPAS Property 3, the results stored in register of indexes I_2^6 is OPT$[[n],0]$.

Notice that optimal permutation $\pi^*[[n],0]$ can be *rebuilt* with registers of indexes I_1^3, I_1^5 and I_1^6, and with the access to the results of the classical part in the QRAM.