Novel Technique for Robust Optimal Algorithmic Cooling

Sadegh Raeisi,[1,*] Mária Kieferová,[2,3,4] and Michele Mosca[2,5,6,7]

[1]*Department of Physics, Sharif University of Technology, Tehran 1458889694, Iran*
[2]*Institute for Quantum Computing, University of Waterloo, Ontario N2L 3G1, Canada*
[3]*Department of Physics and Astronomy, University of Waterloo, Ontario N2L 3G1, Canada*
[4]*Department of Physics and Astronomy, Macquarie University, Sydney, New South Wales 2109, Australia*
[5]*Department of Combinatorics and Optimization, University of Waterloo, Ontario N2L 3G1, Canada*
[6]*Perimeter Institute for Theoretical Physics, Waterloo, Ontario N2L 2Y5, Canada*
[7]*Canadian Institute for Advanced Research, Toronto, Ontario M5G 1Z8, Canada*

Heat-bath algorithmic cooling provides algorithmic ways to improve the purity of quantum states. These techniques are complex iterative processes that change from each iteration to the next and this poses a significant challenge to implementing these algorithms. Here, we introduce a new technique that on a fundamental level, shows that it is possible to do algorithmic cooling and even reach the cooling limit without any knowledge of the state and using only a single fixed operation, and on a practical level, presents a more feasible and robust alternative for implementing heat-bath algorithmic cooling. We also show that our new technique converges to the asymptotic state of heat-bath algorithmic cooling and that the cooling algorithm can be efficiently implemented; however, the saturation could require exponentially many iterations and remains impractical. This brings heat-bath algorithmic cooling to the realm of feasibility and makes it a viable option for realistic application in quantum technologies.

Many quantum effects and quantum technologies rely on fragile quantum fluctuations that can easily be overwhelmed by thermal fluctuations. This is why often techniques for suppressing thermal fluctuations such as cooling in a cryostat or a dilution fridge are required. There are, however, dynamical cooling techniques that more surgically extract energy from subsystems of interest and can lower the temperature beyond what would be feasible with conventional cooling of the entire system.

Heat-bath algorithmic cooling (HBAC) are techniques that operate on an ensemble of qubits and effectively cool down and purify a target subset of qubits. HBAC drives the system out of equilibrium by transferring the entropy from the target qubits to the rest of the ensemble. This is often referred to as "compression" since it uses information theoretical techniques to compress the entropy to the nontarget elements of the ensemble and effectively cools down the target qubits. The target and the refrigerant qubits are often referred to as the "computation" and the "reset" qubits, respectively.

HBAC can be seen as an extension of techniques like dynamic nuclear polarization (DNP) or Insensitive nuclei enhanced by polarization transfer (INEPT) [1] to situations where there is access to more than two spin species and thus could in principle go beyond the purity of the reset qubit. While applications of HBAC go beyond a specific implementation like nuclear magnetic resonance (NMR), it could be combined with techniques known in each

implementation; e.g., DNP can be used to provide the source polarization of HBAC in NMR.

Algorithmic cooling was first introduced in [2] for a closed system, for which, the cooling is limited by the Shannon bound for information compression. This process heats up the reset qubits beyond their initial temperature. It was later proposed to use a heat bath to recycle the reset qubits and enhance the cooling beyond the Shannon bound [3]. In this setting, the reset qubits, through the interaction with a heat bath, are cooled down to the heat-bath temperature again. This is known as the "reset step."

Similar settings have been investigated in the context of quantum thermodynamics (QT) and dynamic cooling [4–9]. The cooling limit, the corresponding resource theories, and generalizations of the third law of thermodynamic are among topics that have attracted a lot of attention in QT [5,7–11]. It turns out that some of these results, like the existence of a cooling limit, apply to HBAC too. Interestingly, even with the help of a heat bath, it is not possible to extract all the entropy from the target qubits [12]. The optimal technique was introduced by Schulman *et al.* in [12] and is known as the partner pairing algorithm (PPA). The existence of the limit was proved by Schulman *et al.* in [12]. Later Raeisi and Mosca [13] established the asymptotic limit of PPA, with the corresponding asymptotic state, and proved that the process asymptotically approaches the cooling limit. We refer to the optimal asymptotic cooling state as OAS.

One of the main challenges with HBAC techniques, especially the ones that converge to OAS, is that they are highly complex. The operations change from each iteration to the next and in many cases, there is no recipe for implementing the operations in each step.

For instance, PPA sorts the diagonal elements of the density matrix in each iteration. But this means not only that one needs to know the state in each iteration, but also that the operation for implementing the sort would change in each iteration as the state changes.

An obvious question is whether or not it would be possible to reach the OAS and the cooling limit with a fixed state-independent operation in each iteration. Note that the state is constantly changing through the cooling process, and naturally, the compression should change too, as is the case with PPA. Reaching the OAS with a fixed operation seems even more nontrivial. In other words, for a HBAC technique with fixed operation, the compression should be tuned such that without knowing the state, not only can it extract entropy from the state, but through the repetition of the process, it would push the state into the OAS. In the language of QT, this translates to finding a periodic evolution that would make an optimal cyclic cooling process. The typical complex nonperiodic evolutions of the HBAC process make it challenging to draw direct connections between QT and HBAC. The existence of a cyclic HBAC technique with fixed iteration would go a long way in bridging this gap.

Here, we answer this question and show that this is in fact possible. We introduce a compression operation that can push the state to the OAS and reach the cooling limit of HBAC and makes a state-independent cycle process. Further, we show that it can be implemented efficiently and give a recipe for building the quantum circuit.

Besides the fundamental significance, this result could have a critical impact on the feasibility of HBAC techniques. First, in contrast to techniques such as PPA, our algorithm can be efficiently implemented. We, however, show that reaching the OAS would require exponentially many iterations. Second, the state independence of operations makes our algorithm simple and more robust and turns HBAC to a viable option for generating large scale supplies of high-purity quantum states.

We start by introducing our algorithm and then present the complexity analysis. Next, we compare it against PPA. We then investigate the robustness of the two techniques.

We assume an ensemble of $n + 1$ qubits, with the first n as the computation and the last as the reset qubits. We use the subscript R and C to refer to the reset and the computation qubits. We also assume that the Hilbert space is ordered as $\mathcal{H}_C \otimes \mathcal{H}_R$; the first part is the computation qubits and the last part is the reset qubits.

In our technique, instead of sorting the diagonal elements, we apply the following unitary for compression in each iteration:

$$
U_{\text{TS}} = \begin{pmatrix} 1 & & & & \\ & X & & & \\ & & \ddots & & \\ & & & X & \\ & & & & 1 \end{pmatrix}, \tag{1}
$$

where X is the Pauli X operator and the first and the last elements of the matrix are one. The matrix is $2^{n+1} \times 2^{n+1}$ and acts on both the computation and the reset qubits. We refer to the unitary U_{TS} as the two-sort operator and to our technique as two-sort algorithmic cooling (TSAC). The unitary U_{TS} swaps every two neighboring elements on the diagonal of the density matrix, except for the first and the last elements. Intuitively, this is a partial sort that acts locally on the density matrix. This is the golden operation that makes it possible to reach the cooling limit without knowing the state.

After compression, the reset qubit is reset which is equivalent to $\mathcal{R}[\rho] = \text{Tr}_R(\rho) \otimes \rho_R$, where Tr_R is the partial trace over the reset qubit and ρ_R is the "reset state"

$$
\rho_R = \frac{1}{z} \begin{pmatrix} e^{\epsilon} & 0 \\ 0 & e^{-\epsilon} \end{pmatrix}, \tag{2}
$$

with $z = (e^{\epsilon} + e^{-\epsilon})$. The parameter ϵ is called the polarization. Our method does not make any nontrivial assumption about ϵ nor n.

Mathematically, each iteration applies the following channel on the full density matrix

$$
\mathcal{C}_{\text{TS}}[\rho] = U_{\text{TS}}^{\dagger}[\text{Tr}_R(\rho) \otimes \rho_R]U_{\text{TS}}. \tag{3}
$$

This process is clearly independent of the iteration or the state and it can be described by a time-homogeneous Markov process. We find the transfer matrix and use its spectrum to prove that the process converges to the OAS and to give an upper bound for the required number of iterations.

The sequences of the elements on the diagonal of the density matrix form a Markov chain. We use a vector with 2^{n+1} elements $\{\lambda^t\}$ to represent the state after the tth iteration. We use a similar notation for the density matrix of the computation qubits (without the reset qubit) and use $\{p^t\}$ instead.

Figure 1 gives a pictorial description of the process in each iteration. It starts with the sequence $\{\lambda^t\}$, the diagonal elements of the density matrix of the n computation, and one reset qubit in the tth iteration. First, there is the reset step which takes the reset qubit to the state in Eq. (2). This takes every two neighboring elements λ_{2k+1}^t and λ_{2k+2}^t to $p_k^t = \lambda_{2k+1}^t + \lambda_{2k+2}^t$ and then splits them into $\zeta_{2k+1}^t = p_k^t e^{\epsilon}/z$ and $\zeta_{2k+2}^t = p_k^t e^{-\epsilon}/z$. Now the two-sort unitary is
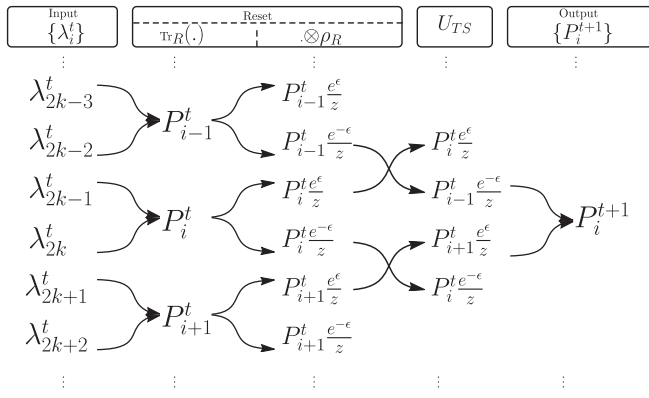
FIG. 1. The pictorial description of an iteration. The input is the list of the diagonal elements of the full density matrix, $\{\lambda^t\}$. The reset step first merges every two neighboring elements (partial trace) and then replaces the reset qubit with ρ_R which splits each element into two elements again. Next, the U_{TS} swaps all the neighboring elements except for the first and the last one.

applied and rearranges the array to $\{\lambda^{t+1}\}$ such that $\lambda^{t+1}_{2k} = \zeta^t_{2k+1}$ and $\lambda^{t+1}_{2k+1} = \zeta^t_{2k}$.

For simplicity, we focus on the computation qubits and trace out the reset qubit. This gives the following update rule for the diagonal elements of the computation qubits:

$$p^{t+1}_i = p^t_{i-1}\frac{e^{-\epsilon}}{z} + p^t_{i+1}\frac{e^{\epsilon}}{z}. \qquad (4)$$

for $1 < i < 2^n$.

Similarly, for the first and the last elements, the update rules are $p^{t+1}_1 = (p^t_1 + p^t_2)(e^{\epsilon}/z)$ and $p^{t+1}_{2^n} = (p^t_{2^n-1} + p^t_{2^n})(e^{-\epsilon}/z)$.

These update rules give the following $2^n \times 2^n$ transition matrix for the Markov process:

$$T = \frac{1}{z}\begin{pmatrix} e^{\epsilon} & e^{\epsilon} & 0 & \cdots & 0 \\ e^{-\epsilon} & 0 & e^{\epsilon} & \cdots & 0 \\ 0 & e^{-\epsilon} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & \ddots & \vdots \\ 0 & 0 & \cdots & e^{-\epsilon} & e^{-\epsilon} \end{pmatrix}. \qquad (5)$$

It is easy to verify that $\{\vec{p^{t+1}}\} = T\{\vec{p^t}\}$ and gives the update rules above. The matrix T has a unique eigenvalue 1 and the remaining eigenvalues are $\Upsilon_k = \{[2\cos(k\pi/2^n)]/z\}$ for $k = 1, 2, \ldots, 2^n - 1$. The eigenstate corresponding to eigenvalue one is

$$\rho = p_0\{1, e^{-2\epsilon}, e^{-4\epsilon}, \ldots\}, \qquad (6)$$

which is the OAS and p_0 is the normalization factor [13]. For the detailed calculation of the eigensystem, see the Supplemental Material (SM) [14]. Since all the other

eigenvalues lie in the interval $(1, -1)$, the Markov chain asymptotically converges to ρ. This proves that our technique asymptotically achieves the cooling limit of HBAC.

We give a $O(n^2)$ circuit for the implementation of the two-sort unitary in the SM [14]. We first shift the basis by one, which transforms it to $\text{Toff}(n + 1)\sigma_x^{(n+1)}$. Then a Pauli σ_x on the reset qubit turns the matrix to a multiple-control Toffoli gate (see SM [14] for details). This shows that our technique can be efficiently implemented. However, to reach the OAS, we need to investigate how many iterations would be required. The mixing time of a Markov chain is the number of iterations required to get within distance ξ of the asymptotic state (i.e., to achieve $|\rho^t - \rho_{OAS}| \leq \xi$). We can upper bound this number of iterations as a function of the spectral gap Δ, i.e., the difference between 1 and the second largest eigenvalue,

$$t_{mix}(\xi) \leq \log\left(\frac{1}{\xi l}\right)\frac{1}{\Delta}, \qquad (7)$$

where $l = p_0 e^{-(2^n-1)\epsilon}$ is the smallest element of the array in Eq. (6) [17].

The spectral gap is $\Delta = 1 - [2\cos(\pi/2^n)]/(e^{\epsilon} + e^{-\epsilon}) \geq [(z-2)/z]$. This gives

$$t_{mix}(\xi) \leq \left[\log\left(\frac{1}{\xi p_0 e^{-(2^n-1)\epsilon}}\right)\left(\frac{z}{z-2}\right)\right]. \qquad (8)$$

It is easy to check that $t_{mix} \in O(2^n)$ which yields $O(n^2 2^n)$ for the overall complexity of TSAC.

Now, we compare our technique with PPA. The key ingredient of PPA is sorting the diagonal elements of the density matrix in each iteration. This transfers as much entropy as possible from the computation elements to the reset qubit [12]. Then the reset qubit is reset back to its equilibrium state.

Even assuming that finding the operation for sorting a 2^{n+1} array is easy, we need to find quantum circuits to implement them which has at least $O(e^n)$ classical complexity. Note that this is only the classical cost of the algorithm. Without this, it would not be even possible to start implementing PPA. For systems as large as 20–40 qubits, e.g., the experiment in [18], not only is it challenging to implement PPA, but it also seems difficult to find the required permutations. This is in contrast to our technique, where each iteration is already known and there is a specific circuit for implementing it.

Next is the gate complexity of PPA. Typically only the number of iterations is counted, ignoring the complexity of the sort operations. In fact, due to the complexity of the sort operations, it is difficult to bound the number of gates required for PPA. Naively, there are $(2^n)!$ permutation matrices of size 2^n. Assuming a finite number of one- and two-qubit gates, there are only $[\text{poly}(n)]^d$ circuits with at
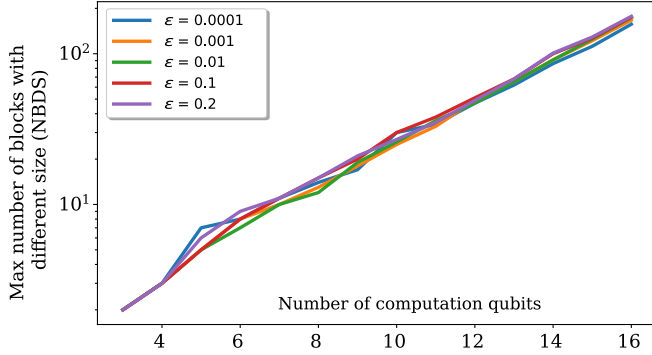
FIG. 2. This plot shows that the maximum of the NBDS for implementations of PPA grows exponentially with the number of computation qubits n. The y axis is a logarithmic scale. Different plots correspond to different reset polarizations ϵ.
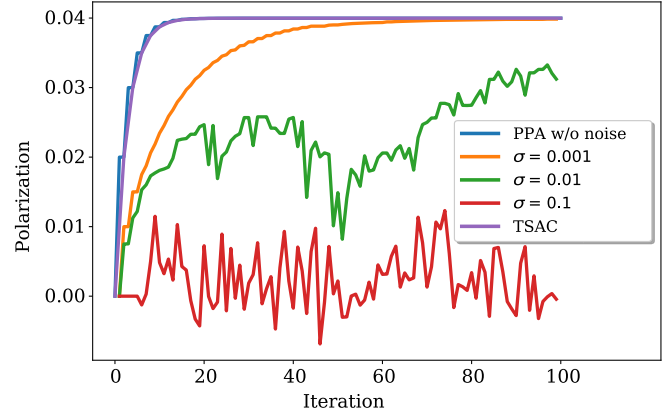

FIG. 3. This plot shows the polarization of the first qubit vs iterations. This is for the simulation of PPA for two computation qubits and one reset qubit, with different amounts of the state estimation errors σ.

most d gates on n qubits. Taking d to be poly(n), one can see that only a small fraction of permutations can be implemented efficiently. Here, we provide a more rigorous bound on the gate complexity of PPA.

The permutation operations can be decomposed into separate cycles that form disjoint blocks in the permutation matrix. These cycles could have different sizes and the size of each cycle determines the number of states it permutes cyclically. These are known as k cycles, where k is the size of the block.

Assume that for all the k cycles in the permutation, we can find an efficient circuit. Also assume that, given a certain state $|j\rangle$, it is possible to efficiently determine which block the state belongs to. Furthermore, we assume that cycles of equal size can be implemented in parallel efficiently. These assumptions may not be true, but any lower bound established with these assumptions still holds when any of these assumptions are weakened or dropped. Under these assumptions, the cost would depend on the number of k cycles with distinct k values. This is the number of blocks in the permutation matrix that have a different size. We refer to this quantity as NBDS.

Implementation of each sort operation requires the implementation of all the blocks. Blocks of different size cannot be fully parallelized and for switching between each of two blocks of unequal size, some quantum operation would be required. This sets the number of blocks of different size NBDS as a lower bound for the complexity of any sort operation.

Figure 2 shows the simulation results of PPA for a different number of computation qubits n and indicates that NBDS grows exponentially with n. This implies that our lower bound for the gate complexity of PPA scales exponentially with n. Here, for any value of n, we get a sequence of permutation matrices and pick the permutation that has the largest NBDS.

Last, there is the fragility to practical imperfections. The sort operation of PPA requires the ordering of the diagonal elements of the state. This means that techniques like

quantum state tomography are needed to monitor the state. This process, however, cannot be perfect and usually there are estimation errors. Figure 3 shows how sensitive the process is to these imperfections. These simulations are for HBAC with $n = 2$ and one reset qubit and the reset polarization of $\epsilon = 0.02$ with a zero-mean Gaussian noise with variance σ. As σ increases, the process becomes random and would not approach the cooling limit any more. Figure 3 also shows the result for TSAC which, regardless of the noise, would always converge to the OAS. This is not a generic noise model, but is relevant for techniques like PPA and shows that with noise, PPA can heat the state. Next, we show that TSAC monotonically pushes the state towards the OAS.

Theorem 1.—Given some state ρ and a reset state ρ_R with polarization ϵ, if the polarization of the first computation qubit is less than the HBAC limit, each iteration of TSAC, as in Eq. (3) would increase the polarization of the first qubit.

Proof.—The polarization of the first qubit is determined by the first half of the diagonal elements of ρ and we need to show that TSAC would increase it. The assumption that the state is hotter than the HBAC limit means

$$p_{2^{n-1}}e^{-\epsilon} < p_{2^{n-1}+1}e^{\epsilon}. \tag{9}$$

After the iteration, these two are swapped by the U_{TS}. This increases the sum of the first half of the diagonal elements and as a result, the polarization of the first qubit.

Note that this can be extended to other computation qubits. It is just easier to show for the first qubit.

Since the process does not depend on the state, even when the state is perturbed from the ideal one, the process continues to cool it down. Note that this does not imply that our algorithm is robust to all imperfections. Specifically, with faulty operations, no algorithm can guarantee the convergence to OAS.

TABLE I. Comparison between PPA and our technique. Here, the "noise sensitivity" is the sensitivity to deviations from the expected state in the process.

	PPA	TSAC
Classical cost of circuit synthesis	$\Omega(2^n)$	Poly (n)
Total number of gates	(Conjecture) $\Omega(2^n)$	$O(n^2 2^n)$
Noise sensitivity	Sensitive	Robust

Table I gives a comparison between TSAC and PPA. Here, the "noise sensitivity" is the sensitivity to deviations from the expected state in the process. The table demonstrates that TSAC outperforms PPA in almost every aspect and presents a more realistic option in practice.

In conclusion, our work presents a novel viable technique for optimal HBAC which shows that optimal HBAC is possible without any knowledge of the state and without changing the operation through the process. From a QT viewpoint, it means that the optimal cooling is possible to do HBAC with a cyclic process. This opens new avenues for examining HBAC in terms of QT. For instance, resources required for reaching the cooling limit have been extensively investigated in QT [5,7–11]. It is interesting to map the required resources and their scaling to HBAC.

Our work also brings realistic applications of these techniques to the realm of possibility. The new technique, in contrast to PPA, uses a fixed operation in every iteration, which addresses the fragility issues in previous works. More precisely, the new technique is robust against imperfections and noise in the state.

It is also possible to combine our work with other dynamics cooling techniques to further reduce the costs. However, it remains open to see how far the complexity may be reduced. Results from QT on the analysis of the resources for cooling and the extensions of the third law of thermodynamics [5,7–9] could prove helpful for reducing the complexity of cost.

[*] sadegh.raeisi@gmail.com

[1] L. T. Kuhn and Ü. Akbey, *Hyperpolarization Methods in NMR Spectroscopy* (Springer, New York, 2013), Vol. 338.

[2] L. J. Schulman and U. V. Vazirani, in *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing* (ACM, New York, 1999), pp. 322–329.

[3] P. O. Boykin, T. Mor, V. Roychowdhury, F. Vatan, and R. Vrijen, Proc. Natl. Acad. Sci. U.S.A. **99**, 3388 (2002).

[4] D. Janzing, P. Wocjan, R. Zeier, R. Geiss, and T. Beth, Int. J. Theor. Phys. **39**, 2717 (2000).

[5] J. Scharlau and M. P. Mueller, Quantum **2**, 54 (2018).

[6] P. Boes, J. Eisert, R. Gallego, M. P. Müller, and H. Wilming, Phys. Rev. Lett. **122**, 210402 (2019).

[7] A. Streltsov, H. Kampermann, S. Wölk, M. Gessner, and D. Bruß, New J. Phys. **20**, 053058 (2018).

[8] L. Masanes and J. Oppenheim, Nat. Commun. **8**, 14538 (2017).

[9] A. E. Allahverdyan, K. V. Hovhannisyan, D. Janzing, and G. Mahler, Phys. Rev. E **84**, 041109 (2011).

[10] D. Reeb and M. M. Wolf, New J. Phys. **16**, 103011 (2014).

[11] C. Browne, A. J. P. Garner, O. C. O. Dahlsten, and V. Vedral, Phys. Rev. Lett. **113**, 100603 (2014).

[12] L. J. Schulman, T. Mor, and Y. Weinstein, Phys. Rev. Lett. **94**, 120501 (2005).

[13] S. Raeisi and M. Mosca, Phys. Rev. Lett. **114**, 100404 (2015).

[14] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevLett.122.220501 for details of the mixing time of the Markov process, implementation costs of TSAC and further details on the effect of the noise, which includes Refs. [15,16].

[15] M. Saeedi and M. Pedram, Phys. Rev. A **87**, 062318 (2013).

[16] T. Beth and M. Rötteler, in *Quantum Information* (Springer, New York, 2001), pp. 96–150.

[17] D. A. Levin, Y. Peres, and E. L. Wilmer, *Markov Chains and Mixing Times* (American Mathematical Society, Providence, 2009).

[18] V. R. Pande, G. Bhole, D. Khurana, and T. S. Mahesh, Phys. Rev. A **96**, 012330 (2017).