The first part of our proof closely follows the optimal upper bound of [WY19] for the support size estimation problem, using Chebyshev polynomial approximations. A labeling scheme for a graph problem can be viewed as a distributed data structure in which all queries must be answered without inspecting the underlying graph, but only the labels of the query arguments. By a standard scaling and rounding argument, one can reduce a problem with real weights and W aspect ratio to the same problem with integer weights and $W\varepsilon^{-1}$ aspect ratio. We now show the correctness of our algorithm. Note that our algorithm is similar to [LP20], but we need to argue its correctness a bit more carefully since whenever we compute an A,B min-cut for a bi-partition, we set the capacities of the edges incident to the source and sink vertices as $\tau + 1$ (instead of ∞ as in their setting). Unfortunately, the proof of Lemma 1.5 turns out to be quite technical. While in the positioning step working with integers had several advantages, in the masking step, we have to deal with several drawbacks. Structural results on the transition graph may in addition have implications for the running time of metaheuristics. In this section we provide a lower bound for the Hamiltonian learning task we consider. Specifically we consider the dependence on the accuracy ϵ and the number of Pauli terms M. The adaptive experiments are modeled as in [60], where all experiments form a tree such that the outcome of the experiment at a vertex determines which child leaf to move to, thus determining the next experiment to perform. Our main tool for the total evolution time lower bound is Assouad's lemma [61], which provides a lower bound on the achievable ℓ_1-error given two prerequisites: (1) an estimate of how hard it is to distinguish two output probability distributions if they come from two Hamiltonians that differ slightly, and (2) a lower bound on the penalty in ℓ_1-error if such a pair of Hamiltonians are not correctly distinguished. More precisely, the algorithm of [KPV24] approximates an object called the "pseudosnapshot", a version of the snapshot with some noise added that adds some errors to the biases of the vertices, to account for the fact that a small change in the bias of a high-degree vertex could substantially change the counts in the snapshot if it happened to be near a border.