**3 A Series of Quantum Algorithms with Varying Complexity**

We provide three quantum algorithms with increasing algorithmic code complexity and decreasing complexity with respect to resource consumption.

**3. 1. A First Attempt**

Before presenting the algorithm, we need to explain some quantum states (M and b(\*, \*) values are as in the algorithm in section 2):

The algorithm, now, goes as follows:

α = n!; STATES = {}

**repeat**

| y> = 1/sqrt(α) Σ\_{u, B} | t\_u> **χ** | B>

Measure the superposed states and let the resulting state be:



where **χ** and **Ε** denotes tensor product and inclusion in set, respectively, u **Ε** U = {u | u = k\_1 + k\_2 + ... +k\_M <= M and k\_j **Ε** {0, 1, ..., M}}, B **Ε** μ\_{((n - 1) x M)} , μ\_{ij} = 0, 1, 1 <= i <= n - 1 and 1 <= j <= M, and **Π** denotes composition of transpositions.

**if** γ\_{(u, B)} is not in STATES **then**

S\_n = S\_n U **Π**\_ν(A^{β\_{1,1}}\_ν Α^{β\_{2,1}}\_ν ... A^{β\_{n - 1,ν}}\_{(n - 1)})^{τ^ν\_u}, (1 <= ν <= Μ)

STATES = STATES U {γ\_{(u, B)}}

α = α - 1

**end**

**until** α = 0

Now #B = 2^{(n - 1)\*M}, #U = o(2^M) and O(M^M) (# denotes "number of"). We activate #B \* #U states totally, that is, o(2^{n \* M}) and O((2^n \* M)^M), and the length of qubit strings is o(n \* M) and O(n \* M + MlogM).