SILQCS - Simple Interpretive Language for Quantum Circuit Simulation ver. 1.1.2

SAITOH Akira*
Graduate School of Engineering Science, Osaka University, JAPAN

November 7, 2004

Abstract

This document describes the specification of an interpretative simulation language for quantum circuit simulation and the usage of the interpreter for it. This software borrows the matrix library of the GAMMA NMR library [1] and the GNU read-line library. The source file is available at `http://www.qc.ee.es.osaka-u.ac.jp/~saitoh/silqcs/`. This software is released under the terms of GNU General Public License version 2.0.

Contents

*E-mail: saitoh _at_ qc.ee.es.osaka-u.ac.jp

[1]GAMMA was written by Dr. Scott A. Smith and Tilo Levante, National High Magnetic Field Laboratory, USA (`http://gamma.magnet.fsu.edu/`) [S.A. Smith, T.O. Levante, B.H. Meier, and R.R. Ernst, J. Magn. Reson., 106a, 75-105, (1994)].

1 Name and synopsis

Language name

SILQCS - Simple Interpretive Language for Quantum Circuit Simulation

Executable file name

qcs

Synopsis

qcs [file]

1.1 File input and *std* input

When a file is specified as

./qcs file

, then the file and the files included from it is interpreted by qcs. On the other hand, when no file name is specified (just ./qcs), it runs in the read-line mode in which the interpreter executes each line input by a user step by step.

2 Grammar

2.1 The style of the simulation language, comment, define, etc.

1. The labels that point qubits are 1,2,...,n from the upper qubits to the lower in the n-qubit input of a circuit.
2. The labels for the ket vectors are $|0\rangle, |1\rangle, \ldots, |2^n - 1\rangle$ for an n-qubit circuit. (Attaching the labels to qubits in the subscripts, the vectors are denoted as $|0_1 0_2 \ldots 0_n\rangle, \ldots, |1_1 1_2 \ldots 1_n\rangle$.)
3. Comment lines that one does not want to execute may be written with '#' in the head of the line:

   # This is a comment.
4. In the file-input mode, each command must be terminated with ';'. The semicolon may be omitted in the read-line mode.
5. In the file-input mode, a line may be concatenated with the next line with a backslash '\'.
6. An external file may be included by

   include "file"

7. A time step of operations is the sentences from ':' to ':'. Before the first colon, the initial vector or matrix must be set by "mload" or a "init..." command.
   Example:

   ```
   init-thermal-biases 0.7 -0.7 0.2;
   10: H(1 2);
   20: showb;
   ```

8. *** You cannot use "define" in the read-line mode. It may be used only in a file. *** The word "define" defines the first subsequent word as the rest subsequent term down to the end of the current line (lines may be concatenated with a backslash). For an example,

   ```
   define SQRT2  1.4142135
   define PI  3.141516
   ```

   Note that a semicolon should not be placed in if you define a constant. A set of operators (or a function) can also be defined *e.g.*

   ```
   define Op1 \
   X(2-5); H(2-5);\
   print This is Op1.;\
   Rx(1-3, 45deg);
   ```

   defines the user-defined operator Op1.

2.2 Iteration

*** The iteration command "times" can be used only in the file-input mode. *** To iterate a block of operations t times (t is an integer >0), command

> t.times{
> *...block of operations...*
> }

Example:

```
10: 3.times{
11: X(3 5 7); 8.times{H(2-8);}
12: print abcdefg;
}
```

3 Set an initial state for pure-qubit computation

3.1 *Init* with binary digits

Suppose there are binary digits $x_i \in \{0,1\}$. One can set the initial ket vector of a pure quantum register to $|x_1 x_2 \ldots x_n\rangle$ by the command

> init x_1 x_2 x_3 ... x_n;

For instance, "init 0 1 0 1;" set the initial state to $|0101\rangle$.

3.2 *Init* with complex magnitudes of each qubit

Suppose that one needs to have an initial state

$$|\psi_{ini}\rangle = \bigotimes_{i=1}^{n} (\alpha_i|0_i\rangle + \beta_i|1_i\rangle) \tag{1}$$

$(\alpha_i, \beta_i \in \mathbf{C}, |\alpha_i|^2 + |\beta_i|^2 = 1)$. This can be done by the command

init $(\alpha_1)(\beta_1)$ $(\alpha_2)(\beta_2)$ ... $(\alpha_n)(\beta_n)$;

where each α_i or β_i is written in the style of "a+bi" or "r ej θ", *e.g.*,

init (i)(0) (0)(1ej0.25pi) (0.7071)(0.7071);

set the initial state to

$$\begin{pmatrix} i \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ e^{i0.25\pi} \end{pmatrix} \otimes \begin{pmatrix} 0.7071 \\ 0.7071 \end{pmatrix}. \tag{2}$$

Note that "pi" (this means π) or "deg" (this means degree) may be used only in the argument θ of an ascii string in the style of "r ej θ".

3.3 *Init* with probabilities of individual qubits being 0

Suppose that one needs to have an initial state

$$|\psi_{ini}\rangle = \bigotimes_{i=1}^{n} (\sqrt{p_i}|0_i\rangle + \sqrt{1-p_i}|1_i\rangle), \tag{3}$$

where $0 \le p_i \le 1, p_i \in \mathbf{R}$. This can be done by the command

initpb p_1 p_2 ... p_n;

For instance, "initpb 0.6 0.5;" sets the initial state to $\begin{pmatrix} \sqrt{0.6} \\ \sqrt{0.4} \end{pmatrix} \otimes \begin{pmatrix} \sqrt{0.5} \\ \sqrt{0.5} \end{pmatrix}$.

4 Set an initial density operator for ensemble quantum computation

4.1 *Init* with probabilities in a thermal state

One can set the initial state to

$$\rho_{ini} = \begin{pmatrix} p_1 & 0 \\ 0 & 1-p_1 \end{pmatrix} \otimes \cdots \otimes \begin{pmatrix} p_n & 0 \\ 0 & 1-p_n \end{pmatrix} \tag{4}$$

$(0 \le p_i \le 1)$ with the command

init-mixed p_1 p_2 ... p_n;

Example in the read-line mode:

```
>init-mixed 0.7 0.6;
>show;
0.42|0><0|+ 0.28|1><1|+ 0.18|2><2|+ 0.12|3><3|
>CN(1,2);
>show;
0.42|0><0|+ 0.28|1><1|+ 0.12|2><2|+ 0.18|3><3|
```

4.2 *Init* with biases in a thermal state

One can set the initial state to

$$\rho_{ini} = \begin{pmatrix} \frac{1+\varepsilon_1}{2} & 0 \\ 0 & \frac{1-\varepsilon_1}{2} \end{pmatrix} \otimes \cdots \otimes \begin{pmatrix} \frac{1+\varepsilon_n}{2} & 0 \\ 0 & \frac{1-\varepsilon_n}{2} \end{pmatrix} \tag{5}$$

$(0 \leq \varepsilon_i \leq 1)$ with the command

init-thermal-biases ε_1 ε_2 ... ε_n;

4.3 Specify a matrix

To use the specific matrix

$$\begin{pmatrix} c_{0\ 0} & c_{0\ 1} & \cdots & c_{0\ 2^n-1} \\ c_{1\ 0} & & & \\ \vdots & & \ddots & \\ c_{2^n-2\ 0} & & \cdots & \\ c_{2^n-1\ 0} & c_{2^n-1\ 1} & \cdots & c_{2^n-1\ 2^n-1} \end{pmatrix} \tag{6}$$

as the initial state, command "init-mixed" with a subsequent matrix:

init-mixed $(c_{0\ 0}\ c_{0\ 1}\ \ldots\ c_{0\ 2^n-1})(c_{1\ 0}\ \ldots\ )$ ... $(c_{2^n-1\ 0}\ c_{2^n-1\ 1}\ \ldots\ c_{2^n-1\ 2^n-1})$;

c_i may be written in the style of "0.2+0.3i" or "0.4ej0.25pi". We recommend "mload" command rather than this invocation.

5 Save and load the current vector or density matrix

You can save the current state into a file with the command

msave *file*;

You can load the current state from a file with the command

mload *file*;

The matrix

$$\begin{pmatrix} c_{0\ 0} & c_{0\ 1} & \cdots & c_{0\ 2^n-1} \\ c_{1\ 0} & & & \\ \vdots & & \ddots & \\ c_{2^n-2\ 0} & & \cdots & \\ c_{2^n-1\ 0} & c_{2^n-1\ 1} & \cdots & c_{2^n-1\ 2^n-1} \end{pmatrix} \tag{7}$$

is saved as the text

$(c_{0\ 0}\ c_{0\ 1}\ \ldots\ c_{0\ 2^n-1})(c_{1\ 0}\ \ldots\ )\ldots(c_{2^n-1\ 0}\ c_{2^n-1\ 1}\ \ldots\ c_{2^n-1\ 2^n-1});$

5.1 Save a matrix to a temporary file

If a temporary file is preferred, the command

mtmp *file*;

may be used to save the current state. This *file* is held in a directory for temporary files specified by the environmental variable "SILQCS_TMP" (/tmp is the default directory), which is removed when a user exits from *silqcs*. The actual name of the temporary file looks "`silqcs-_~-`" + *file*. A user can use the saved temporary file by specifying its name just *file* instead of its actual path and filename.

The directory in which a temporary file is saved may be changed also by the command

settmpdir *directory*;

where *directory* is a path.

5.2 Delete a file

A user can delete his/her file by the command

mfdel *file*;

This command is not recommended because it can delete any file of a user.

5.3 Matrix file conversion into Octave, Maxima, and GP/PARI

The following commands convert the matrix file *src* created by "msave", "RDOsave", or "PTsave" into the file *dst* with the format of Octave, Maxima, and GP/PARI respectively.

conv-octave (*src*, *dst*);

conv-maxima (*src*, *dst*);

conv-gp (*src*, *dst*);

Example:

The matrix file *mfile0*

```
(0 0 0 0 0 0 0 0)(0 -2.439696962e-01 4.200000000e-01+4.200000000e-01i 0 0 0 0 0)
(0 4.200000000e-01-4.200000000e-01i 9.439696962e-01 0 0 0 0 0)(0 0 0 0 0 0 0 0)(
0 0 0 0 0 0 0 0)(0 0 0 0 0 -1.045584412e-01 0 -1.800000000e-01-1.800000000e-01i)
(0 0 0 0 0 0 0 0)(0 0 0 0 0 -1.800000000e-01+1.800000000e-01i 0 4.045584412e-01)
```

is converted by the command

conv-octave(mfile0, mfile0.octave);

to the file *mfile0.octave*

```
[0,0,0,0,0,0,0,0;0,-2.439696962e-01,4.200000000e-01+4.200000000e-01i,0,0,0,0,0;0
,4.200000000e-01-4.200000000e-01i,9.439696962e-01,0,0,0,0,0;0,0,0,0,0,0,0,0;0,0,
0,0,0,0,0,0;0,0,0,0,0,-1.045584412e-01,0,-1.800000000e-01-1.800000000e-01i;0,0,0
,0,0,0,0,0;0,0,0,0,0,-1.800000000e-01+1.800000000e-01i,0,4.045584412e-01]
```

6 Intrinsic operations

6.1 Single-qubit operations

6.1.1 General single-qubit unitary operation

This command is used to apply a given 1-qubit operation to the qubits specified by *bit labels*. Format:

U (*bit labels*, (a b)(c d));

where bit labels are given by the text in the style of "1-6" (qubit 1 to qubit 6) and/or "1 3 5" (qubit 1, 3, and 5), and (a b)(c d) is regarded as the matrix $\begin{pmatrix} \mathrm{a} & \mathrm{b} \\ \mathrm{c} & \mathrm{d} \end{pmatrix}$. a,...,d may be complex numbers. Example:

U (1 2 4-7, (0 -i)(i 0));

When a user defines a matrix A by "matrix A (a b)(c d);" (See also Sec. 7.1) prior to the invocation of U(...), then

U(*bit labels*, A);

may be used. However, in this case, one should use

A(*bit labels*);

The matrix may be an intrinsic matrix "X", "Rx(angle)", etc., *e.g.*,

U(1-3, Rx(45deg));

6.1.2 Pauli matrices, rotation, etc.

* The "*bit labels*" is a string in the style of "1-6" (qubit 1 to qubit 6) and/or "1 3 5" (qubit 1, 3, and 5).

1. Pauli matrices
   (a) NOT (*bit labels*);
   (b) X (*bit labels*);
   (c) Y (*bit labels*);
   (d) Z (*bit labels*);
2. Hadamard gate
   (a) H (*bit labels*);
3. Rotation Here, "angle" is the angle in radian (*e.g.* 0.78539816). To specify it in degree, add "deg" to the decimal numbers (*e.g.* 45deg).
   (a) Rx (*bit labels*, angle);
   (b) Ry (*bit labels*, angle);
   (c) Rz (*bit labels*, angle);

6.2 Controlled-U

Format:

C (*control bit labels*, *target bit labels*, *matrix*);

The *control bit labels* is a string in the style of "1-6" (qubit 1 to qubit 6) and/or "1 o3 5" (qubit 1, o3, and 5). Here, "o" is used to specify a 0-control bit, *i.e.,* "o3" means that the *matrix* is applied to the target bits under the condition that the bit 3 is 0 besides the condition of the other control bits. The *target bit labels* is a string in the style of "1-6" (qubit 1 to qubit 6) and/or "1 3 5" (qubit 1, 3, and 5). The *matrix* may be given in the style of (a b)(c d) or a user-defined matrix "A" or something. "X", "Y", "Z", "H", "Rx(angle)", "Ry(angle)", "Rz(angle)" may be used as a matrix to apply to the targets. "angle" is the angle in radian (*e.g.* 0.78539816). To specify it in degree, add "deg" to the decimal numbers [*e.g.* C(1, 3, Rx(45deg))].

6.3 CNOT

Format:

CN (*control bit labels*, *target bit labels*);

The *control bit labels* is a string in the style of "1-6" (qubit 1 to qubit 6) and/or "1 o3 5" (qubit 1, o3, and 5). Here, "o" is used to specify a 0-control bit, *i.e.,* "o3" means that the X gate is applied to the target bits under the condition that the bit 3 is 0 besides the condition of the other control bits. The *target bit labels* is a string in the style of "1-6" (qubit 1 to qubit 6) and/or "1 3 5" (qubit 1, 3, and 5).

6.4 SWAP, Toffoli, Fredkin gates

Here, a,b, and c are integers between 0 and n.

1. SWAP gate:

   SW (a, b); or SW (a b);

2. Toffoli gate with the control bits a and b:

   Tf (a b, c);

3. Toffoli gate with the control bits b and c:

   Tf (a, b c);

4. Fredkin gate with the SWAPped bits b and c:

   Fr (a, b c);

5. Fredkin gate with the SWAPped bits a and b:

   Fr (a b, c);

6.5 Quantum Fourier Transformation

The QFT:

FT (*bit labels*);

The QFT^{-1}:

FTˆ-1 (*bit labels*);

* The "*bit labels*" is a string in the style of "1-6" (qubit 1 to qubit 6) and/or "1 3 5" (qubit 1, 3, and 5).

6.6 Projective measurement with projectors $P_m = |m\rangle\langle m|$

Note: Measurement has not yet been implemented for a mixed state. It works only for a pure state kept as a vector.

An available type of observable for a measurement in this software is currently only the type: $M = \sum_m mP_m$ with $P_m = |m\rangle\langle m| = |x_a x_b x_c ...\rangle\langle x_a x_b x_c ...|$ where $x_a x_b x_c ...$ is a number written by using binary numbers ($x_i \in \{0, 1\}$) and $|x_a x_b x_c ...\rangle$ are computational basis vectors of qubits whose labels are specified by a user.

1. To specify the binary values to be measured (measurement result) (0 or 1) for measured qubits, use the command:

   M (*bit labels*, x_a x_b x_c ...);

   * The "*bit labels*" is a string in the style of "1-6" (qubit 1 to qubit 6) and/or "1 3 5" (qubit 1, 3, and 5). x_a x_b ... is the string of the binary values to be measured for the qubits of "*bit labels*". You may type like this:

   M (1-8, 1 0 0 1 0 1 1 1);

   The qubits that are not measured will properly transit to the state after the measurement.

   Example for the read-line mode:

   ```
    > init 0 0;
    > H(1); CN(1,2);
    > showb;
   0.707107|00> + 0.707107|11>
    > M(1,0);
    > showb;
   1|00>
   ```

2. When you do not specify the result of measurement, the measured qubits result in 0 or 1 with the corresponding probabilities. Such a transition is controlled by a random seed that is produced from the current local time of a computer. For an example, you may try the above example with the line

   M(1);

   instead of `M(1,0);`.

7 Non-intrinsic operations

7.1 User-defined 1-qubit operation

"matrix" command is acceptable both in the file-input mode and in the read-line mode; this is unlike the command "define" in the item 8 of Sec. 2.1. You may use your own matrix for 1-qubit operation by using the "matrix" command.

matrix A (a b)(c d);
matrix A=(a b)(c d);

Once you define it, it can be invoked as

A(*bit labels*);

* The "*bit labels*" is a string in the style of "1-6" (qubit 1 to qubit 6) and/or "1 3 5" (qubit 1, 3, and 5).

8 Noisy channel

In this software, only the independent error model is handled. In the model, a error in a wire is independent of those in other wires. Here is a command:

IEOpE (*bit labels*, $\underset{E_1}{(a \;\; b)(c \;\; d)}$, $\underset{E_2}{(e \;\; f)(g \;\; h)}$, ...);

This command applies a single-wire noisy channel with operation elements E_1, E_2, ... to every qubits specified by *bit labels*. An operation element must be written in the form "$(a \;\; b)(c \;\; d)$" such as "$(0 \;\; 0.1i)(-0.1i \;\; 0)$".

9 Visibility

The following command changes the mode of the machine to the *visibility mode* in which the visibility of a unitary transformation can be calculated:

visibility mode;

The visibility of a quantum state σ with respect to a unitary transformation U is generally defined as

$$v = |\mathrm{Tr}(U\sigma)|. \tag{8}$$

No sooner the command "visibility mode" is input than the current state is copied to the state σ. The subsequent quantum gates U is applied to σ without $U^\dagger$ as well as to ρ with $U^\dagger$ (ρ is the state for ordinary calculation), that is to say, both $U\sigma$ and $U\rho U^\dagger$ are calculated internally after that command. To see the visibility, one types the command:

visibility;

There are other commands related to the visibility. "vtrace" command is to show the matrix $\mathrm{Tr}(U\sigma)$; "vphase" shows the phase $\mathrm{Arg}\{\mathrm{Tr}(U\sigma)\}$.

10 Reduced density operator

RDO(*bit labels*);

outputs the reduced density operator of the qubits specified by *bit labels*. To show each vector in the output of the reduced density operator in the format of binary digits, you may command

RDOb(*bit labels*);

Similarly,

RDOsave (*bit labels*, *file name*);

outputs the reduced density operator of the qubits specified by *bit labels* to the file of *file name*. Further analysis with the saved matrix should be performed in another calculator such as Octave. This is possible when a user converts it to the matrix in another format (See Sec. 5.3).

To save the reduced density operator of specified qubits into a temporary file, use the command

RDOtmp (*bit labels*, *file name*);

See also Sec. 5 for details about temporary files.

Note: These four commands does not change the current state in calculation. They only show or save a reduced density operator.

In general, a reduced density operator of specified qubits $(a, b, c, ...)$ is given by

$$\tilde{\rho}_{a,b,c,...} = \sum_{x_{others}} \langle x_{others}|\rho|x_{others}\rangle, \tag{9}$$

where $\{x_{others}\}$ are all the bit strings of the qubits other than $a, b, c, ...$.

Let the component system A comprise the specified qubits $a, b, c, ...$, and the component system B comprise the other qubits. Then the reduced density operator of A is defined as

$$\tilde{\rho}_A = \sum_{\varphi_B} \langle \varphi_B|\rho|\varphi_B\rangle, \tag{10}$$

where $\{|\varphi_B\rangle\}$ is a complete orthonormal basis of H_B (the Hilbert space of B).

10.1 Average bipartite entanglement

The command

ABE_each;

outputs the average bipartite entanglement [D. A. Meyer and N. R. Wallach, J. Math. Phys. **43** 4273 (2002)] between each qubit and the rest (n-1)-qubits state [J. Emerson, Y. S. Weinstein, M. Saraceno, S. Lloyd, and D. G. Cory, Science **302** 2098 (2003)],

$$Q = 2 - \frac{2}{n}\sum_{i=1}^{n} \mathrm{Tr}(\tilde{\rho}_i^2), \tag{11}$$

where $\tilde{\rho}_i$ is the reduced density operator of the i-th qubit.

WARNING: This is NOT a good measure of entanglement for mixed states, especially those with initial small polarizations. It has been hoped that some good measure of entanglement for mixed states will be proposed in some day.

11 Partial transpose

PT (*bit labels*);

outputs the partial transpose with respect to the qubits specified by *bit labels*. There is also the command

PTb (*bit labels*);

This shows each vector of the output of the partial transpose in the format of binary digits.

To save the partial transpose into the file named *file name*, use the command:

PTsave (*bit labels*, *file name*);

The saved matrix can be loaded by "mload;". Further analysis with the saved matrix should be performed in another calculator such as Octave. This is possible when a user converts it to the matrix in another format (See Sec. 5.3).

To save the partial transpose into a temporary file, use the command

PTtmp (*bit labels*, *file name*);

See also Sec. 5 for details about temporary files.

Note: These four commands does not change the current state in calculation. They only show or save a partial transpose.

In general, the partial transpose of a density operator ρ of a bipartite state (the state space is the Hilbert space $H_A \otimes H_B$) with respect to the component system A (H_A) is denoted as ρ^{T_A}. Let us write the i-j element of ρ as

$$\rho_{ij} = \langle \varphi_i | \rho | \varphi_j \rangle = \langle \varphi_{i,A} | \langle \varphi_{i,B} | \rho | \varphi_{j,A} \rangle | \varphi_{j,B} \rangle. \tag{12}$$

Then the i-j element of ρ^{T_A} is given by

$$\rho_{ij}^{T_A} = \langle \varphi_{j,A} | \langle \varphi_{i,B} | \rho | \varphi_{i,A} \rangle | \varphi_{j,B} \rangle. \tag{13}$$

In other words, the *ac*-*bd* element of ρ, $\langle a|\langle c|\rho|b\rangle|d\rangle$, is mapped to the *bc*-*ad* element of ρ^{T_A}, $\langle b|\langle c|\rho^{T_A}|a\rangle|d\rangle$, for all the possible integers a, b, c, d, where $|a\rangle, |b\rangle$ are basis vectors of the component system A, and $|c\rangle, |d\rangle$ are basis vectors of the component system B.

For example, the partial transpose with respect to the left most qubit of $\rho = \frac{1}{2}(|010\rangle\langle 010| + |010\rangle\langle 111| + |111\rangle\langle 010| + |111\rangle\langle 111|)$ is $\rho^{T_l} = \frac{1}{2}(|010\rangle\langle 010| + |011\rangle\langle 110| + |110\rangle\langle 011| + |111\rangle\langle 111|)$. For this purpose, you may just type "PTb(1);" after setting up the mixed state ρ. Try the following example in the read-line mode:

```
======================
 > init-mixed 1.0 0.0 1.0;
 > H(3); CN(3,1);
 > showb;
0.5|010><010| + 0.5|010><111| + 0.5|111><010| + 0.5|111><111|
 > PTb(1);
0.5|010><010| + 0.5|011><110| + 0.5|110><011| + 0.5|111><111|
 > showb;
0.5|010><010| + 0.5|010><111| + 0.5|111><010| + 0.5|111><111|
 >
```

12 Eigenvalues

12.1 Display eigenvalues

The command

eig;

shows all the eigenvalues of the current density matrix.

The command

eigRDO(*bit labels*);

shows all the eigenvalues of the reduced density operator of the qubits specified by *bit labels*.

The command

eigPT(*bit labels*);

shows all the eigenvalues of the partial transpose with respect to the qubits specified by *bit labels*.

12.2 Display the von Neumann entropy

Generally, the von Neumann entropy of a density matrix ρ is defined by

$$S_{vN}(\rho) = -\mathrm{Tr}\rho \log_2 \rho. \tag{14}$$

We use the binary logarithm (base 2) in this software. It is also written by using the eigenvalues $\{\lambda_i\}$ of ρ as

$$S_{vN}(\rho) = -\sum_i \lambda_i \log_2 \lambda_i. \tag{15}$$

This is because a density matrix ρ can be diagonalized by a proper unitary transformation. Such a transformation preserves eigenvalues of ρ. A density matrix ρ is an Hermitian matrix and its eigenvalues are non-negative real numbers.

The command

SvN;

shows the von Neumann entropy of the current density matrix.

The command

SvNRDO(*bit labels*);

shows the von Neumann entropy of the reduced density operator of the qubits specified by *bit labels*.

13 Distance measures

13.1 Trace distance

The command below calculates the trace distance between the two matrices A and B read from respective files of *fileA* and *fileB* saved by the "msave" command (or "RDOsave" etc.).

trdist(*fileA*, *fileB*);

The trace distance between (N,N) density matrices A and B is defined by

$$\begin{aligned} D(A,B) &= \frac{1}{2}\mathrm{Tr}|A-B| \\ &= \frac{1}{2}\mathrm{Tr}\sqrt{(A-B)^\dagger(A-B)}. \end{aligned}$$

13.2 Fidelity

The command below calculates the fidelity between the two matrices A and B read from respective files of *fileA* and *fileB* saved by the "msave" command (or "RDOsave" etc.).

fidelity(*fileA*, *fileB*);

The fidelity between (N,N) density matrices A and B is defined by

$$F(A,B) = \mathrm{Tr}\sqrt{A^{\frac{1}{2}}BA^{\frac{1}{2}}}. \tag{16}$$

13.3 Umegaki's relative entropy

The command "REnt" calculates the Umegaki relative entropy of a matrix A with respect to a matrix B,

$$S(A\|B) = \mathrm{Tr}(A\log_2\frac{A}{B}). \tag{17}$$

A and B are read from the two matrix files *fileA* and *fileB* saved by the "msave" command (or "RDOsave" etc.).

REnt(*fileA*, *fileB*);

14 Print, show, etc.

14.1 Print the current state

1. Decimal representation

   show (*basis vector labels*);

   shows the component states (the complex amplitudes and the basis vectors) of specified labels in the style of $0.7071|5\rangle$ if the current state is kept as a vector. When the current state is represented as a density matrix, this command shows all the non-zero elements in the style of $0.5|3\rangle\langle 5|$, ignoring the *basis vector labels*.

   show;

   shows all the non-zero elements of the current state in the decimal representation.

2. Binary representation

   showb (*basis vector labels*);

   shows the component states (the complex amplitudes and the basis vectors) of specified labels in the style of $0.7071|0101\rangle$ if the current state is kept as a vector. When the current state is represented as a density matrix, this command shows all the non-zero elements in the style of $0.5|0011\rangle\langle 0101|$, ignoring the *basis vector labels*.

   showb;

   shows all the non-zero elements of the current state in the binary representation.

14.2 Print a joint probability

The joint probability of qubits specified by "*bit labels*" being in the state specified by binary numbers "$x_a \;\; x_b \;\; x_c \;\; ...$" is output to the *stdout* by the command

prob (*bit labels*, $x_a \;\; x_b \;\; x_c \;\; ...$);

For instance,

prob (1 3-5 8 9, 0 1 0 0 1 0);

shows the joint probability $\Pr(0_1 1_3 0_4 0_5 1_8 0_9)$ (subscripts are the labels of qubits).

14.3 Print probabilities or biases (polarizations) of individual qubits

The commands below internally use the same routine as that for calculating reduced density operators (See Sec. 10). Let the reduced density operator of the i-th qubit be $\tilde{\rho}_i = \begin{pmatrix} a_i & b_i \\ b_i^* & 1-a_i \end{pmatrix}$. Then the probability of the i-th qubit being 0 is a_i, and the superficial bias of it is defined as ${}^s\varepsilon_i = 2a_i - 1$. The values of a_i and ${}^s\varepsilon_i$ may be changed even by single-qubit rotation gates. On the other hand, there is a definition of polarization that is independent of a macroscopic spin direction: Let ${}^{d}\tilde{\rho}_i = \begin{pmatrix} \lambda_i & 0 \\ 0 & 1-\lambda_i \end{pmatrix}$ be the matrix obtained from a unitary diagonalization of $\tilde{\rho}_i$ ($\lambda_i \geq 0.5$). Then the intrinsic bias of the qubit is defined as ${}^{i}\varepsilon_i = 2\lambda_i - 1$.

1. Every probability of each specified qubit being measured as 0 is printed by the command:

   bprobs (*bit labels*);

2. All the probabilities of individual qubits being measured as 0 are printed by the command:

   bprobs;

3. The intrinsic biases of specified individual qubits are printed by the command:

   ibiases (*bit labels*);

4. All the intrinsic biases of individual qubits are printed by the command:

   ibiases;

5. The superficial biases of specified individual qubits are printed by the command:

   sbiases (*bit labels*);

6. All the superficial biases of individual qubits are printed by the command:

   sbiases;

Example:

```
 > init-mixed 0.6 0.7 0.8;
 > bprobs(1-3);
1:0.6 2:0.7 3:0.8
 > sbiases;
1:0.2 2:0.4 3:0.6
 > H(2);
 > sbiases;
1:0.2 2:0 3:0.6
 > ibiases;
1:0.2 2:0.4 3:0.6
```

14.4 Print the total binary entropy

TBE (*bit labels*);

shows the sum of the binary entropies of individual qubits specified by *bit labels*. A binary entropy of the i-th qubit is

$$H_b(\varepsilon_i) = -\left(\frac{1+\varepsilon_i}{2}\log_2\frac{1+\varepsilon_i}{2} + \frac{1-\varepsilon_i}{2}\log_2\frac{1-\varepsilon_i}{2}\right), \qquad (18)$$

where ε_i is the intrinsic polarization of the i-th qubit, which is calculated according to the eigenvalues, λ_i and $1-\lambda_i$, of the reduced density operator

$\tilde{\rho}_i$ of the qubit. In other words, the command shows the total binary entropy

$$S_b = \sum_{specified \quad i} S_{vN}(\tilde{\rho}_i), \tag{19}$$

where S_{vN} denotes the von Neumann entropy.

14.5 "print" command

Print the *string* to *stdout*.

print *string*;

"*string*" may contain the escape characters "\a" (beep), "\t" (tab), "\n" (new line), etc.

14.6 Precision in outputs

The precision of values output into *stdout* may be changed with the command

normal (*precision*);

Here, *precision* is an integer. A user may also get a scientifically output values, *e.g.* 1.0e-3, in all outputs. For this purpose, type the command

scientific (*precision*);

When the specified precision is larger than 9, it is also applied to the outputs of "msave", "RDOsave", "conv-octave", etc. By default, the values of matrix elements are output with the precision of 9 in the scientific style.

15 Output the circuit image

Still unstable...

16 Program examples

The examples of quantum circuit simulation program are placed in the directory of "examples" in the package of silqcs. Here we show two simple examples.

16.1 3-pure-qubit Grover's iteration

The first example is 3-pure-qubit Grover's iteration [L.K. Grover, Proc. 28th STOC pp.212-219, quant-ph/9605043 (1996)].

Program:

```
init 0 0 0 1;

10: H(1-4);
20: print Find |010> from the haymow state, (H x H x H)|000>\n;
```

```
30: print Initially, Pr(010)=;
40: prob(1 2 3, 0 1 0);

50: 7.times{
##### Grover's iteration #####
51: CN(o1 2 o3, 4);
52: H(1-3);
53: CN(o1 o2 o3, 4);
54: H(1-3);
55: print At this time, Pr(010)=;
56: prob(1 2 3, 0 1 0);
##############################
}
```

The result from the run of this program:

```
[akira@localhost silqcs]$ ./qcs examples/Grover.q
Find |010> from the haymow state, (H x H x H)|000>
Initially, Pr(010)=0.125
At this time, Pr(010)=0.78125
At this time, Pr(010)=0.945312
At this time, Pr(010)=0.330078
At this time, Pr(010)=0.012207
At this time, Pr(010)=0.547974
At this time, Pr(010)=0.999786
At this time, Pr(010)=0.576973
```

16.2 In-place addition using QFT

The second example is the in-place addition using QFT [A. Fijany and C.P. Williams, quant-ph/9809004 (1998); T.G. Draper, quant-ph/0008033 (2000)].

Program:

```
# The initial state is \rho=diag(1.0, 0.0) x diag(1.0, 0.0) x diag(0.0, 1.0).
# In the command of init-mixed, a user specifies every
# population of |0> of an individual qubit.

init-mixed 1.0 1.0 0.0;

# Definition of extrinsic matrices. Here, ej means \exp(i) and pi means \pi.
matrix R1 (1 0)(0 1ej1pi);
matrix R2 (1 0)(0 1ej0.5pi);
matrix R3 (1 0)(0 1ej0.25pi);

# The beginning of the circuit.
0: print Initial state is \t; showb; print Then apply H(2).\n;
10: H(2);

20: print The state before the addition circuit is\n; showb;
30: print Add 001 using QFT.\n;

# Apply QFT to the bits 1-3.
40: FT(1-3);
# Attach the phases, i.e., R3 to bit 3, R2 to bit 2, R1 to bit 1.
50: R3(3); R2(2); R1(1);
# Apply QFT^-1 to the bits 1-3.
60: FT^-1(1-3);

70: print The result is\n;
80: showb;
```

The result from the run of this program:

```
[akira@localhost silqcs]$ ./qcs examples/Inplace-Add-Density-Op.q
Initial state is        1|001><001|,
Then apply H(2).
The state before the addition circuit is
0.5|001><001| + 0.5|001><011| + 0.5|011><001| + 0.5|011><011|
Add 001 using QFT.
The result is
0.5|010><010| + 0.5|010><100| + 0.5|100><010| + 0.5|100><100|
```

17 Frequently Asked Questions

17.1 Warning message in calculating eigenvalues or in calculating the von Neumann entropy

The intrinsic operations "eig", "eigRDO", "eigPT", "SvN", and "SvNRDO" sometimes output the following message:

```
Warning: could not find eigenvalues correctly.
```

This means that *silqcs* could not achieve a sufficient accuracy in calculating eigenvalues. Internally, a sort of Hermitian Jacobi method is used for the eigenvalue calculation in *silqcs*. This method is known to be successful for any Hermitian matrix. However, some elements of a matrix kept in a computer memory may have small numerical errors owing to rounding off or owing to the low precision of standard C++ functions. Thus, it is possible to lose an accuracy in calculating eigenvalues. In this case, a user has to use another external calculator such as Octave to calculate eigenvalues or the von Neumann entropy. (Note that even a famous mathematical calculator may fail in keeping a sufficient accuracy in calculating eigenvalues for such a matrix that causes the above warning message.)

17.2 How to use an external calculator

We will see how to use an external calculator. For example, suppose you have a reduced density operator $\tilde{\rho}_{1,2}$ in the read-line mode:

```
 > init-mixed 0.7 0.7 0.9;
 > H(1 2);CN(1 2,3);
 > RDO(1 2);
0.25|0><0| + 0.1|0><1| + 0.1|0><2| + 0.1|1><0| + 0.25|1><1| +
0.04|1><2| + 0.1|2><0| + 0.04|2><1| + 0.25|2><2| + 0.25|3><3|
 > SvNRDO(1 2)
1.87811
```

As we have seen above, the von Neumann entropy of $\tilde{\rho}_{1,2}$ is $S_{vN}(\tilde{\rho}_{1,2}) = 1.87811$.

This matrix can be saved and converted to the matrix in the style of Octave by the commands:

```
 > RDOsave(1 2, foo.matrix);
 > conv-octave(foo.matrix, foo.matrix.octave);
```

The file "foo.matrix.octave" contains the text:

```
[2.500000000e-01,1.000000000e-01,1.000000000e-01,0;1.000000000e-01,2.500000000e-
01,4.000000000e-02,0;1.000000000e-01,4.000000000e-02,2.500000000e-01,0;0,0,0,2.5
00000000e-01]
```

Then you can calculate the von Neumann entropy $S_{vN}(\tilde{\rho}_{1,2})$ also by using Octave. The Octave function to calculate $S_{vN}(\rho)$ for a density matrix ρ is

```
%%%% The Octave function to calculate the von Neumann entropy of rho %%%%
function S = SvN(rho)
   A = eig(rho);
```

```
   S = 0;
   for i=1:1:rows(A)
      if A(i,1) ~= 0,
         S += A(i,1)*log(A(i,1))/log(2);
      endif
   endfor
   S = -S;
endfunction
```

This function and the rest part of the program

```
A=[2.500000000e-01,1.000000000e-01,1.000000000e-01,0;1.000000000e-01,2.500000000e
-01,4.000000000e-02,0;1.000000000e-01,4.000000000e-02,2.500000000e-01,0;0,0,0,2.5
00000000e-01];
SvN(A)
```

returns the result $S_{vN}(\tilde{\rho}_{1,2}) = 1.8781$ as shown below.

```
[akira@localhost transition]$ octave foo.octave
GNU Octave, version 2.1.35 (i386-redhat-linux-gnu).
Copyright (C) 1996, 1997, 1998, 1999, 2000, 2001 John W. Eaton.
This is free software with ABSOLUTELY NO WARRANTY.
For details, type `warranty'.

ans = 1.8781
```

This value is equal to the one obtained by *silqcs*.