**Classic Genetic Algorithm**

Diagram

Description automatically generated

Problems:

A graph with numbers and letters

Description automatically generated with low confidenceA *mobile entity* (ME) can discretely move on a sphere.

ME has to reach some *target points* Tg= {(xga,yga), …, (xgb,ygb)} avoiding some *trap points* Tp= {(xga,yga), …, (xgb,ygb)}.

When ME reaches a point *(i,j),* the cost is:

ME model:

*[O(τ+1),X(τ+1),Y(τ+1)]= SH(τ )∙[O(τ),X(τ),Y(τ)]* where

* *O(τ)* represents the orientation at the moment *τ*
  + 0 for right (O(0) = 0, as initial orientation),
  + 1 for down,
  + 2 for left and
  + 3 for up.
* *X(τ),Y(τ)]* is the position at the moment *τ*, with *[X(0),Y(0)]= [1,1]*

ME move is represented by the sequence

σ=[O(0),X(0),Y(0)]\* [O(1),X(1),Y(1)]\*[O(2),X(2),Y(2)]\*∙∙∙\*[O(10),X(10),Y(10)].

SH performs a rotation and the shift to the new position. The ME rotation in a point (i,j) is given by the value si,j:

*O(τ+1)* = modulo4(*O(τ) +* ri,j(*τ))* if X(*τ)=i;* Y(*τ)=j*;

Where si,j is an element of the matrix SH.

The operator SH moves ME to the next position after the rotation is performed.

There are two kinds of SH (i.e. shift to the new position):

* ri,j(*τ)* = constant for any *τ;*
* ri,j(*τ)* depends on *τ*.

The move performance is assessed with the formula:

R=[ri,j; 0≤i≤7, 0≤j≤7];

The elements ri,j; 0≤i≤7, 0≤j≤7 determine the rotation value.

Time horizon is 0≤ *τ* ≤16 and the request is .

*Improvement of assessment*

If the sequence σ includes points and such that *[i,j]* precedent to *[u,v]*, then the cost *Cost([u,v]) = 0*.

**Problem 1**

Find R (i.e. *ri,j* is constant for all the time *τ)* that maximizes *Jp*.

**Problem 2**

Find R(*τ)* (i.e. *ri,j(τ)* depend on the time *τ* too)that maximizes *Jp*.

Genome for *Problem 1*: R=[ri,j; 0≤i≤7, 0≤j≤7]; 🡪 64 elements

Genome for *Problem 2*: R(*τ* )=[ri,j(*τ)*; 0≤i≤7, 0≤j≤7, 0≤ *τ* ≤16]; 🡪 64 × 16= 1024 elements.

*Genetic operators*:

Mutation 🡨 change randomly ri,j modulo 4;

Crossover: select two individuals and perform simple cut crossover.

**Quantum Genetic Algorithm**

Classic Genetic Algorithm (GA) has to execute (simulate) many individuals for their assessment one by one.

QC (quantum computing) has the advantage that can simulate simultaneously a large number of individuals.

QC uses quantum variables that can cover by superposition the entire search domain.

QGAs (Quantum Genetic Algorithms) have to represent the individuals by quantum vectors. The individual modifications fulfill the Hilbert condition by using mappings involving *unitary matrices*!

Quantum approach can be used in GA by:

* *Quantum Inspired Genetic Algorithms* (QIGA) where the simulation and improvement are implemented on classical computers;
* Using *Hybrid GA (HGA)* where the simulations are performed by QC, but the improvements are determined by task implemented on classical computer;
* The individual simulations and improvements are performed by tasks implemented *completely on QC* *(CQGA)*.

**QIGA**

Each gene is composed of two qubits. For the current problem, the total number of qubits (i.e. the genome length) is 64 and this imply g=128 qubits.

*Random transformation* (considered genetic operation) on V representation:

*A(θ)* that acts on each qubit. 🡪 The vectors will meet the Hilbert condition because *A(θ)* is unitary.

Vtj an individual ‘j’ in the generation ‘t’ with the element qbti in the position ‘i’.

Transformation of individuals: qbti = A(θk) ∙ qbt-1i

*Initialization:*