G-ROUTING ALGORITHM METHODOLOGY

The **G-ROUTING ALGORITHM Methodology** is invented to provide routing algorithm for the **G-NETWORK** layouts in some geometries like Satellites, Internet **3D-ëWeb Networks**, **Cell ëMap System** and even for **Emergency Traffic Lights Routing System**. The **G-ROUTING ALGORITHM** provides great routing, fast and robust for any networks using **Neighbor-to-Neighbors** networking technology. The Neighbor-to-Neighbors networking technology only requires neighbor knowing next neighbors; one network node only needs to know its closest neighbors and can only communicate to its closest neighbors. The **Neighbor-to-Neighbors** networking technology provides the network with self-adjusting when a node is removed or added into the networks.

The **G-ROUTING ALGORITHM Methodology** is invented with State-of-the-Art and invented to work with any geometry networks, G-NETWORK from **simple networking nodes** to most **complicated machine networking nodes**. The simple networking nodes are the network nodes that can only send and receive signal to forward to the next closest neighbors; these networking nodes do not have any knowledge about the network configuration database or the neighbor nodes' properties. These network nodes cannot perform a re-route when the desired network routing path stuck at their position with a neighbor node is disconnected during routing. A center network controller is required to build a route path with a series of routing nodes and let one node forward to its neighbor routing node until it reaches the destination node. The **complicate machine networking nodes** are the nodes that have knowledge of the network configuration database or their neighbor nodes' properties. These networking nodes can perform a re-route when the desired network routing path stuck at their position with a neighbor node is disconnected during routing. For **complicate machine networking nodes**, any nodes can be used to build and start a network route. The systems with these networking nodes can be built with **Auto-Notify-Neighbors** and broadcasting across entire networks for updating when a node is gone missing or a new node has been added into the network structure. For complex networking like **Star-Tree ëWeb** topology network, network nodes can be mixed with both re-routable and not re-routable nodes; complicated machine nodes and simple forwarding only nodes. The **G-ROUTING ALGORITHM** is a promise for future of networking and routing algorithm with the **Neighbor-to-Neighbors** networking technology.

G-ROUTING ALGORITHM METHODOLOGY

Henry V. Pham

Network Nodes Layouts

Figure-1 below shows the square network nodes layout which is similar to the Satellite StarLink network layout and Traffic Lights network layout. The network nodes are communicated directly to their closest neighbors; and they can only send and receive to their closest neighbor nodes. This layout provides a node can only have maximum of 4-neighbor nodes.

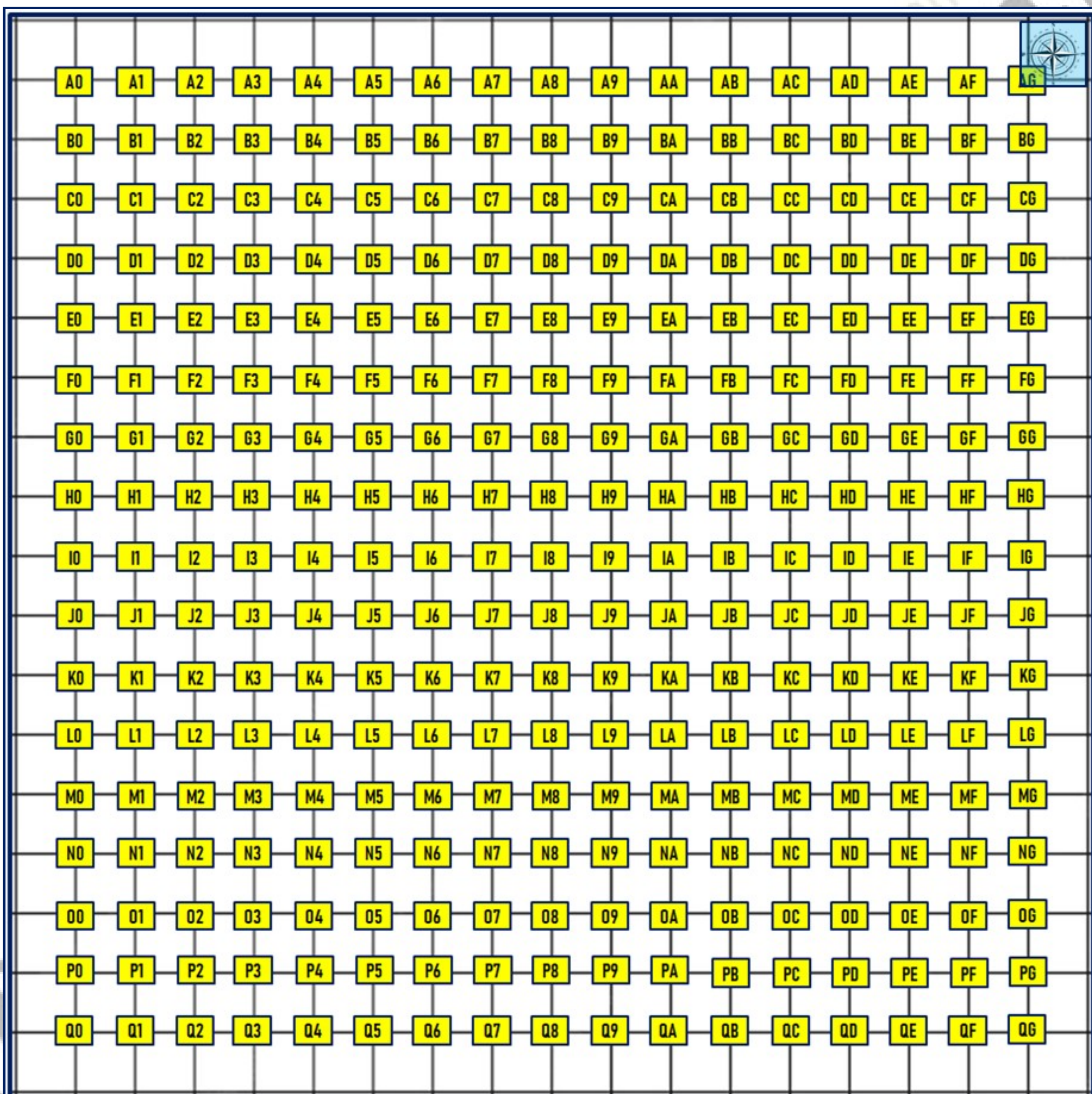


Figure-1: NETWORK NODES IN SQUARE LAYOUT

Figure-2 below shows the square network nodes layout plus diagonal lines which provides networking layout with some shortcut paths for faster routing networking system. The network nodes are crossed by diagonal lines having more neighbor nodes than others. The lines showing in the network layout can be direct connected or wireless in any forms but required reachable node-to-node or Neighbor-to-Neighbors only. The **Neighbor-to-Neighbors** networking technology is not just in square layout, but it can be in hexagon layout or any other layouts like Star-Tree topology layout or can be in **Multi-layer Network Structure** with **local-to-local** nodes and **community-to-community** nodes within the same network.

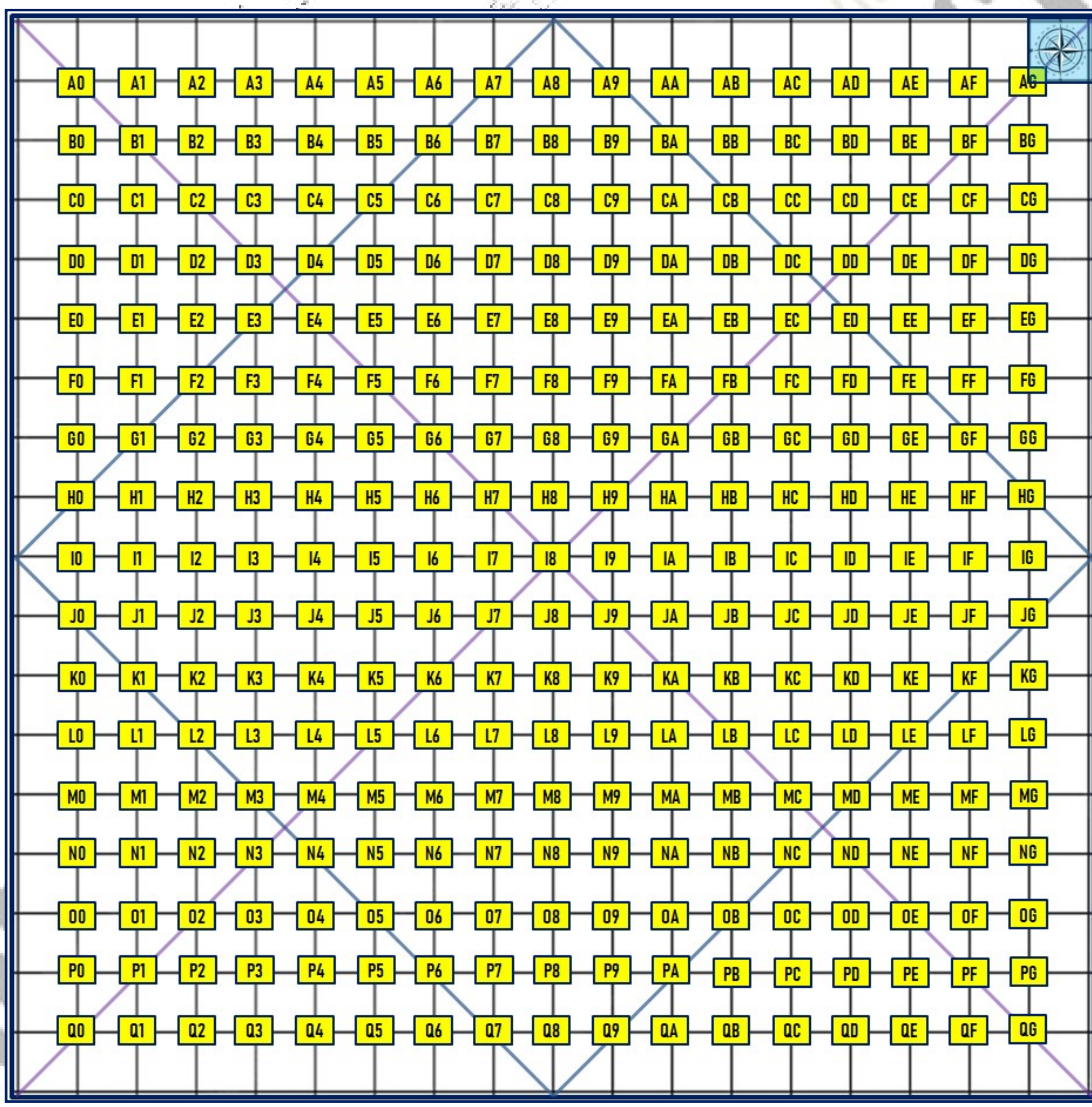


Figure-2: NETWORK NODES IN SQUARE W/ DIAGONALS LAYOUT

G-ROUTING ALGORITHM Database Requirements

Table-1 below shows a simple database for the network layout in **Figure-2** above. Each line shows a node with its list of neighbors and its neighbors' properties with distance and direction from that node. The nodes that are highlight in **orange** are the neighbor nodes on the diagonal lines of the outer square; and nodes that are highlight in **blue** are on the inner square sides. By default for square layout, each node has maximum of 4-neighbor nodes not including the ones in diagonal lines. The table shows node '**B1**' has 6-neighbor nodes in order '**A1; B2; C2; C1; B0; A0**' of clockwise direction start from 12 O' clock. Node '**A1**' has a distance of '**dA1**' in North direction; node '**B2**' has a distance of '**dB2**' in East direction; node '**C2**' has a distance of '**dC2**' in **South East** direction; node '**C1**' has a distance of '**dC1**' in South direction; node '**B0**' has a distance of '**dB0**' in West direction; and node '**A0**' has a distance of '**dA0**' in **North West** direction. This node is a sample node from the table database below; other nodes are having the same format.

The G-ROUTING ALGORITHM will start at a source node and look for the list its neighbor nodes; again, the list of neighbor nodes is recommended to be in order of clockwise start at 12 O' clock. The next sequence of routing will pick the first node from the list and look for this node's neighbors; and pick the next node from the list and look for its neighbor nodes and so on. With the same method, the algorithm will start at a destination node and look for the list its neighbor nodes. When iterating through nodes on each side, duplicate nodes from previous iterations will be eliminated. The algorithm will iterate one node at a time for both source node and destination node until both paths contain a shared node. This shared node will be in the first route which will connect the source and destination nodes together for the first route. The algorithm also provides flexible of more routes to choose by performing more iterations through node-by-node on both source and destination nodes. This algorithm can be written in a recursive method for both source and destination nodes until both paths contain a shared node. Next sections will show more in details and the rules of the G-ROUTING ALGORITHM for the above networking layouts.

NODE	NEIGHBOR NODES	DISTANCE/DIRECTION TO NEIGHBOR NODES (meters)
A0	A1;B1;B0	A1:=[dA1/E]; B1:=[dB1/SE]; B0:=[dB0/S]
A1	A2;B1;A0	A2:=[dA2/E]; B1:=[dB1/S]; A0:=[dA0/W]
A2	A3;B2;A1	A3:=[dA3/E]; B2:=[dB2/S]; A1:=[dA1/W]
A3	A4;B3;A2	A4:=[dA4/E]; B3:=[dB3/S]; A2:=[dA2/W]
A4	A5;B4;A3	A5:=[dA5/E]; B4:=[dB4/S]; A3:=[dA3/W]
A5	A6;B5;A4	A6:=[dA6/E]; B5:=[dB5/S]; A4:=[dA4/W]
A6	A7;B6;A5	A7:=[dA7/E]; B6:=[dB6/S]; A5:=[dA5/W]
A7	A8;B7;B6;A6	A8:=[dA8/E]; B7:=[dB7/S]; B6:=[dB6/SW]; A6:=[dA6/W]
A8	A9;B8;A7	A9:=[dA9/E]; B8:=[dB8/S]; A7:=[dA7/W];
A9	AA;BA;B9;A8	AA:=[dAA/E]; BA:=[dBA/SE]; B9:=[dB9/S]; A8:=[dA8/W]
AA	AB;BA;A9	AB:=[dAB/E]; BA:=[dBA/S]; A9:=[dA9/W]
AB	AC;BB;AA	AC:=[dAC/E]; BB:=[dBB/S]; AA:=[dAA/W]
AC	AD;BC;AB	AD:=[dAD/E]; BC:=[dBC/S]; AB:=[dAB/W]
AD	AE;BD;AC	AE:=[dAE/E]; BD:=[dBD/S]; AC:=[dAC/W]
AE	AF;BE;AD	AF:=[dAF/E]; BE:=[dBE/S]; AD:=[dAD/W]
AF	AG;BF;AE	AG:=[dAG/E]; BF:=[dBF/S]; AE:=[dAE/W]
AG	BG;BF;AF	BG:=[dBG/S]; BF:=[dBF/SW]; AF:=[dAF/W]
B0	A0;B1;C0	A0:=[dA0/N]; B1:=[dB1/E]; C0:=[dC0/S]
B1	A1;B2;C2;C1;B0;A0	A1:=[dA1/N]; B2:=[dB2/E]; C2:=[dC2/SE]; C1:=[dC1/S]; B0:=[dB0/W]; A0:=[dA0/NW]
B2	A2;B3;C2;B1	A2:=[dA2/N]; B3:=[dB3/E]; C2:=[dC2/S]; B1:=[dB1/W]
B3	A3;B4;C3;B2	A3:=[dA3/N]; B4:=[dB4/E]; C3:=[dC3/S]; B2:=[dB2/W]
B4	A4;B5;C4;B3	A4:=[dA4/N]; B5:=[dB5/E]; C4:=[dC4/S]; B3:=[dB3/W]
B5	A5;B6;C5;B4	A5:=[dA5/N]; B6:=[dB6/E]; C5:=[dC5/S]; B4:=[dB4/W]
B6	A6;A7;B7;C6;C5;B5	A6:=[dA6/N]; A7:=[dA7/NE]; B7:=[dB7/E]; C6:=[dC6/S]; C5:=[dC5/SW]; B5:=[dB5/W]
B7	A7;B8;C7;B6	A7:=[dA7/N]; B8:=[dB8/E]; C7:=[dC7/S]; B6:=[dB6/W]
B8	A8;B9;C8;B7	A8:=[dA8/N]; B9:=[dB9/E]; C8:=[dC8/S]; B7:=[dB7/W]
B9	A9;BA;C9;B8	A9:=[dA9/N]; BA:=[dBA/E]; C9:=[dC9/S]; B8:=[dB8/W]
BA	AA;BB;CB;CA;B9;A9	AA:=[dAA/N]; BB:=[dBB/E]; CB:=[dCB/SE]; CA:=[dCA/S]; B9:=[dB9/W]; A9:=[dA9/NW]
BB	AB;BC;CB;BA	AB:=[dAB/N]; BC:=[dBC/E]; CB:=[dCB/S]; BA:=[dBA/W]
BC	AC;BD;CC;BB	AC:=[dAC/N]; BD:=[dBD/E]; CC:=[dCC/S]; BB:=[dBB/W]
BD	AD;BE;CD;BC	AD:=[dAD/N]; BE:=[dBE/E]; CD:=[dCD/S]; BC:=[dBC/W]
BE	AE;BF;CE;BD	AE:=[dAE/N]; BF:=[dBF/E]; CE:=[dCE/S]; BD:=[dBD/W]
BF	AF;AG;BG;CF;CE;BE	AF:=[dAF/N]; AG:=[dAG/NE]; BG:=[dBG/E]; CF:=[dCF/S]; CE:=[dCE/SW]; BE:=[dBE/W]
BG	AG;CG;BF	AG:=[dAG/N]; CG:=[dCG/S]; BF:=[dBF/W]
C0	B0;C1;D0	B0:=[dB0/N]; C1:=[dC1/E]; D0:=[dD0/S]
C1	B1;C2;D1;C0	B1:=[dB1/N]; C2:=[dC2/E]; D1:=[dD1/S]; C0:=[dC0/W]
C2	B2;C3;D3;D2;C1;B1	B2:=[dB2/N]; C3:=[dC3/E]; D3:=[dD3/SE]; D2:=[dD2/S]; C1:=[dC1/W]; B1:=[dB1/NW]
C3	B3;C4;D3;C2	B3:=[dB3/N]; C4:=[dC4/E]; D3:=[dD3/S]; C2:=[dC2/W]
C4	B4;C5;D4;C3	B4:=[dB4/N]; C5:=[dC5/E]; D4:=[dD4/S]; C3:=[dC3/W]
C5	B5;B6;C6;D5;D4;C4	B5:=[dB5/N]; B6:=[dB6/NE]; C6:=[dC6/E]; D5:=[dD5/S]; D4:=[dD4/SW]; C4:=[dC4/W]
C6	B6;C7;D6;C5	B6:=[dB6/N]; C7:=[dC7/E]; D6:=[dD6/S]; C5:=[dC5/W]
C7	B7;C8;D7;C6	B7:=[dB7/N]; C8:=[dC8/E]; D7:=[dD7/S]; C6:=[dC6/W]
C8	B8;C9;D8;C7	B8:=[dB8/N]; C9:=[dC9/E]; D8:=[dD8/S]; C7:=[dC7/W]
C9	B9;CA;D9;C8	B9:=[dB9/N]; CA:=[dCA/E]; D9:=[dD9/S]; C8:=[dC8/W]
CA	BA;CB;DA;C9	BA:=[dBA/N]; CB:=[dCB/E]; DA:=[dDA/S]; C9:=[dC9/W]
CB	BB;CC;DC;DB;CA;BA	BB:=[dBB/N]; CC:=[dCC/E]; DC:=[dDC/SE]; DB:=[dDB/S]; CA:=[dCA/W]; BA:=[dBA/NW]
CC	BC;CD;DC;CB	BC:=[dBC/N]; CD:=[dCD/E]; DC:=[dDC/S]; CB:=[dCB/W]
CD	BD;CE;DD;CC	BD:=[dBD/N]; CE:=[dCE/E]; DD:=[dDD/S]; CC:=[dCC/W]
CE	BE;BF;CF;DE;DD;CD	BE:=[dBE/N]; BF:=[dBF/NE]; CF:=[dCF/E]; DE:=[dDE/S]; DD:=[dDD/SW]; CD:=[dCD/W]
CF	BF;CG;DF;CE	BF:=[dBF/N]; CG:=[dCG/E]; DF:=[dDF/S]; CE:=[dCE/W]
CG	BG;DG;CF	BG:=[dBG/N]; DG:=[dDG/S]; CF:=[dCF/W]

D0	C0;D1;E0	C0:=[dC0/N]; D1:=[dD1/E]; E0:=[dE0/S]
D1	C1;D2;E1;D0	C1:=[dC1/N]; D2:=[dD2/E]; E1:=[dE1/S]; D0:=[dD0/W]
D2	C2;D3;E2;D1	C2:=[dC2/N]; D3:=[dD3/E]; E2:=[dE2/S]; D1:=[dD1/W]
D3	C3;D4;E4;E3;D2;C2	C3:=[dC3/N]; D4:=[dD4/E]; E4:=[dE4/SE]; E3:=[dE3/S]; D2:=[dD2/W]; C2:=[dC2/NW]
D4	C4;C5;D5;E4;E3;D3	C4:=[dC4/N]; C5:=[dC5/NE]; D5:=[dD5/E]; E4:=[dE4/S]; E3:=[dE3/SW]; D3:=[dD3/W]
D5	C5;D6;E5;D4	C5:=[dC5/N]; D6:=[dD6/E]; E5:=[dE5/S]; D4:=[dD4/W]
D6	C6;D7;E6;D5	C6:=[dC6/N]; D7:=[dD7/E]; E6:=[dE6/S]; D5:=[dD5/W]
D7	C7;D8;E7;D6	C7:=[dC7/N]; D8:=[dD8/E]; E7:=[dE7/S]; D6:=[dD6/W]
D8	C8;D9;E8;D7	C8:=[dC8/N]; D9:=[dD9/E]; E8:=[dE8/S]; D7:=[dD7/W]
D9	C9;DA;E9;D8	C9:=[dC9/N]; DA:=[dDA/E]; E9:=[dE9/S]; D8:=[dD8/W]
DA	CA;DB;EA;D9	CA:=[dCA/N]; DB:=[dDB/E]; EA:=[dEA/S]; D9:=[dD9/W]
DB	CB;DC;EB;DA	CB:=[dCB/N]; DC:=[dDC/E]; EB:=[dEB/S]; DA:=[dDA/W]
DC	CC;DD;ED;EC;DB;CB	CC:=[dCC/N]; DD:=[dDD/E]; ED:=[dED/SE]; EC:=[dEC/S]; DB:=[dDB/W]; CB:=[dCB/NW]
DD	CD;CE;DE;ED;EC;DC	CD:=[dCD/N]; CE:=[dCE/NE]; DE:=[dDE/E]; ED:=[dED/S]; EC:=[dEC/SW]; DC:=[dDC/W]
DE	CE;DF;EE;DD	CE:=[dCE/N]; DF:=[dDF/E]; EE:=[dEE/S]; DD:=[dDD/W]
DF	CF;DG;EF;DE	CF:=[dCF/N]; DG:=[dDG/E]; EF:=[dEF/S]; DE:=[dDE/W]
DG	CG;EG;DF	CG:=[dCG/N]; EG:=[dEG/S]; DF:=[dDF/W]
E0	D0;E1;F0	D0:=[dD0/N]; E1:=[dE1/E]; F0:=[dF0/S]
E1	D1;E2;F1;E0	D1:=[dD1/N]; E2:=[dE2/E]; F1:=[dF1/S]; E0:=[dE0/W]
E2	D2;E3;F2;E1	D2:=[dD2/N]; E3:=[dE3/E]; F2:=[dF2/S]; E1:=[dE1/W]
E3	D3;D4;E4;F3;F2;E2	D3:=[dD3/N]; D4:=[dD4/NE]; E4:=[dE4/E]; F3:=[dF3/S]; F2:=[dF2/SW]; E2:=[dE2/W]
E4	D4;E5;F5;F4;E3;D3	D4:=[dD4/N]; E5:=[dE5/E]; F5:=[dF5/SE]; F4:=[dF4/S]; E3:=[dE3/W]; D3:=[dD3/NW]
E5	D5;E6;F5;E4	D5:=[dD5/N]; E6:=[dE6/E]; F5:=[dF5/S]; E4:=[dE4/W]
E6	D6;E7;F6;E5	D6:=[dD6/N]; E7:=[dE7/E]; F6:=[dF6/S]; E5:=[dE5/W]
E7	D7;E8;F7;E6	D7:=[dD7/N]; E8:=[dE8/E]; F7:=[dF7/S]; E6:=[dE6/W]
E8	D8;E9;F8;E7	D8:=[dD8/N]; E9:=[dE9/E]; F8:=[dF8/S]; E7:=[dE7/W]
E9	D9;EA;F9;E8	D9:=[dD9/N]; EA:=[dEA/E]; F9:=[dF9/S]; E8:=[dE8/W]
EA	DA;EB;FA;E9	DA:=[dDA/N]; EB:=[dEB/E]; FA:=[dFA/S]; E9:=[dE9/W]
EB	DB;EC;FB;EA	DB:=[dDB/N]; EC:=[dEC/E]; FB:=[dFB/S]; EA:=[dEA/W]
EC	DC;DD;ED;FC;FB;EB	DC:=[dDC/N]; DD:=[dDD/NE]; ED:=[dED/E]; FC:=[dFC/S]; FB:=[dFB/SW]; EB:=[dEB/W]
ED	DD;EE;FE;FD;EC;DC	DD:=[dDD/N]; EE:=[dEE/E]; FE:=[dFE/SE]; FD:=[dFD/S]; EC:=[dEC/W]; DC:=[dDC/NW]
EE	DE;EF;FE;ED	DE:=[dDE/N]; EF:=[dEF/E]; FE:=[dFE/S]; ED:=[dED/W]
EF	DF;EG;EF;EE	DF:=[dDF/N]; EG:=[dEG/E]; FF:=[dFF/S]; EE:=[dEE/W]
EG	DG;FG;EF	DG:=[dDG/N]; FG:=[dFG/S]; EF:=[dEF/W]
F0	E0;F1;G0	E0:=[dE0/N]; F1:=[dF1/E]; G0:=[dG0/S]
F1	E1;F2;G1;F0	E1:=[dE1/N]; F2:=[dF2/E]; G1:=[dG1/S]; F0:=[dF0/W]
F2	E2;E3;F3;G2;G1;F1	E2:=[dE2/N]; E3:=[dE3/NE]; F3:=[dF3/E]; G2:=[dG2/S]; G1:=[dG1/SW]; F1:=[dF1/W]
F3	E3;F4;G3;F2	E3:=[dE3/N]; F4:=[dF4/E]; G3:=[dG3/S]; F2:=[dF2/W]
F4	E4;F5;G4;F3	E4:=[dE4/N]; F5:=[dF5/E]; G4:=[dG4/S]; F3:=[dF3/W]
F5	E5;F6;G6;G5;F4;E4	E5:=[dE5/N]; F6:=[dF6/E]; G6:=[dG6/SE]; G5:=[dG5/S]; F4:=[dF4/W]; E4:=[dE4/NW]
F6	E6;F7;G6;F5	E6:=[dE6/N]; F7:=[dF7/E]; G6:=[dG6/S]; F5:=[dF5/W]
F7	E7;F8;G7;F6	E7:=[dE7/N]; F8:=[dF8/E]; G7:=[dG7/S]; F6:=[dF6/W]
F8	E8;F9;G8;F7	E8:=[dE8/N]; F9:=[dF9/E]; G8:=[dG8/S]; F7:=[dF7/W]
F9	E9;FA;G9;F8	E9:=[dE9/N]; FA:=[dFA/E]; G9:=[dG9/S]; F8:=[dF8/W]
FA	EA;FB;GA;F9	EA:=[dEA/N]; FB:=[dFB/E]; GA:=[dGA/S]; F9:=[dF9/W]
FB	EB;EC;FC;GB;GA;FA	EB:=[dEB/N]; EC:=[dEC/NE]; FC:=[dFC/E]; GB:=[dGB/S]; GA:=[dGA/W]; FA:=[dFA/W]
FC	EC;FD;GC;FB	EC:=[dEC/N]; FD:=[dFD/E]; GC:=[dGC/S]; FB:=[dFB/W]
FD	ED;FE;GD;FC	ED:=[dED/N]; FE:=[dFE/E]; GD:=[dGD/S]; FC:=[dFC/W]
FE	EE;FF;GF;GE;FD;ED	EE:=[dEE/N]; FF:=[dFF/E]; GF:=[dGF/SE]; GE:=[dGE/S]; FD:=[dFD/W]; ED:=[dED/NW]
FF	EF;FG;GF;FE	EF:=[dEF/N]; FG:=[dFG/E]; GF:=[dGF/S]; FE:=[dFE/W]
FG	EG;GG;FF	EG:=[dEG/N]; GG:=[dGG/S]; FF:=[dFF/W]

G0	F0;G1;H0	F0:=[dF0/N]; G1:=[dG1/E]; H0:=[dH0/S]
G1	F1;F2;G2;H1;H0;G0	F1:=[dF1/N]; F2:=[dF2/NE]; G2:=[dG2/E]; H1:=[dH1/S]; H0:=[dH0/SW]; G0:=[dG0/W]
G2	F2;G3;H2;G1	F2:=[dF2/N]; G3:=[dG3/E]; H2:=[dH2/S]; G1:=[dG1/W]
G3	F3;G4;H3;G2	F3:=[dF3/N]; G4:=[dG4/E]; H3:=[dH3/S]; G2:=[dG2/W]
G4	F4;G5;H4;G3	F4:=[dF4/N]; G5:=[dG5/E]; H4:=[dH4/S]; G3:=[dG3/W]
G5	F5;G6;H5;G4	F5:=[dF5/N]; G6:=[dG6/E]; H5:=[dH5/S]; G4:=[dG4/W]
G6	F6;G7;H7;H6;G5;F5	F6:=[dF6/N]; G7:=[dG7/E]; H7:=[dH7/SE]; H6:=[dH6/S]; G5:=[dG5/W]; F5:=[dF5/NW]
G7	F7;G8;H7;G6	F7:=[dF7/N]; G8:=[dG8/E]; H7:=[dH7/S]; G6:=[dG6/W]
G8	F8;G9;H8;G7	F8:=[dF8/N]; G9:=[dG9/E]; H8:=[dH8/S]; G7:=[dG7/W]
G9	F9;GA;H9;G8	F9:=[dF9/N]; GA:=[dGA/E]; H9:=[dH9/S]; G8:=[dG8/W]
GA	FA;FB;GB;HA;H9;G9	FA:=[dFA/N]; FB:=[dFB/NE]; GB:=[dGB/E]; HA:=[dHA/S]; H9:=[dH9/SW]; G9:=[dG9/W]
GB	FB;GC;HB;GA	FB:=[dFB/N]; GC:=[dGC/E]; HB:=[dHB/S]; GA:=[dGA/W]
GC	FC;GD;HC;GB	FC:=[dFC/N]; GD:=[dGD/E]; HC:=[dHC/S]; GB:=[dGB/W]
GD	FD;GE;HD;GC	FD:=[dFD/N]; GE:=[dGE/E]; HD:=[dHD/S]; GC:=[dGC/W]
GE	FE;GF;HE;GD	FE:=[dFE/N]; GF:=[dGF/E]; HE:=[dHE/S]; GD:=[dGD/W]
GF	FF;GG;HG;HF;GE;FE	FF:=[dFF/N]; GG:=[dGG/E]; HG:=[dHG/SE]; HF:=[dHF/S]; GE:=[dGE/W]; FE:=[dFE/NW]
GG	FG;HG;GF	FG:=[dFG/N]; HG:=[dHG/S]; GF:=[dGF/W]
H0	G0;G1;H1;I0	G0:=[dG0/N]; G1:=[dG1/NE]; H1:=[dH1/E]; I0:=[dI0/S]
H1	G1;H2;I1;H0	G1:=[dG1/N]; H2:=[dH2/E]; I1:=[dI1/S]; H0:=[dH0/W]
H2	G2;H3;I2;H1	G2:=[dG2/N]; H3:=[dH3/E]; I2:=[dI2/S]; H1:=[dH1/W]
H3	G3;H4;I3;H2	G3:=[dG3/N]; H4:=[dH4/E]; I3:=[dI3/S]; H2:=[dH2/W]
H4	G4;H5;I4;H3	G4:=[dG4/N]; H5:=[dH5/E]; I4:=[dI4/S]; H3:=[dH3/W]
H5	G5;H6;I5;H4	G5:=[dG5/N]; H6:=[dH6/E]; I5:=[dI5/S]; H4:=[dH4/W]
H6	G6;H7;I6;H5	G6:=[dG6/N]; H7:=[dH7/E]; I6:=[dI6/S]; H5:=[dH5/W]
H7	G7;H8;I8;I7;H6;G6	G7:=[dG7/N]; H8:=[dH8/E]; I8:=[dI8/SE]; I7:=[dI7/S]; H6:=[dH6/W]; G6:=[dG6/NW]
H8	G8;H9;I8;H7	G8:=[dG8/N]; H9:=[dH9/E]; I8:=[dI8/S]; H7:=[dH7/W]
H9	G9;GA;HA;I9;I8;H8	G9:=[dG9/N]; GA:=[dGA/NE]; HA:=[dHA/E]; I9:=[dI9/S]; I8:=[dI8/SW]; H8:=[dH8/W]
HA	GA;HB;IA;H9	GA:=[dGA/N]; HB:=[dHB/E]; IA:=[dIA/S]; H9:=[dH9/W]
HB	GB;HC;IB;HA	GB:=[dGB/N]; HC:=[dHC/E]; IB:=[dIB/S]; HA:=[dHA/W]
HC	GC;HD;IC;HB	GC:=[dGC/N]; HD:=[dHD/E]; IC:=[dIC/S]; HB:=[dHB/W]
HD	GD;HE;ID;HC	GD:=[dGD/N]; HE:=[dHE/E]; ID:=[dID/S]; HC:=[dHC/W]
HE	GE;HF;IE;HD	GE:=[dGE/N]; HF:=[dHF/E]; IE:=[dIE/S]; HD:=[dHD/W]
HF	GF;HG;IF;HE	GF:=[dGF/N]; HG:=[dHG/E]; IF:=[dIF/S]; HE:=[dHE/W]
HG	GG;IG;HF;GF	GG:=[dGG/N]; IG:=[dIG/S]; HF:=[dHF/W]; GF:=[dGF/NW]
I0	H0;I1;J0	H0:=[dH0/N]; I1:=[dI1/E]; J0:=[dJ0/S]
I1	H1;I2;J1;I0	H1:=[dH1/N]; I2:=[dI2/E]; J1:=[dJ1/S]; I0:=[dI0/W]
I2	H2;I3;J2;I1	H2:=[dH2/N]; I3:=[dI3/E]; J2:=[dJ2/S]; I1:=[dI1/W]
I3	H3;I4;J3;I2	H3:=[dH3/N]; I4:=[dI4/E]; J3:=[dJ3/S]; I2:=[dI2/W]
I4	H4;I5;J4;I3	H4:=[dH4/N]; I5:=[dI5/E]; J4:=[dJ4/S]; I3:=[dI3/W]
I5	H5;I6;J5;I4	H5:=[dH5/N]; I6:=[dI6/E]; J5:=[dJ5/S]; I4:=[dI4/W]
I6	H6;I7;J6;I5	H6:=[dH6/N]; I7:=[dI7/E]; J6:=[dJ6/S]; I5:=[dI5/W]
I7	H7;I8;J7;I6	H7:=[dH7/N]; I8:=[dI8/E]; J7:=[dJ7/S]; I6:=[dI6/W]
I8	H8;H9;I9;J9;J8;J7;I7;H7	H8:=[dH8/N]; H9:=[dH9/NE]; I9:=[dI9/E]; J9:=[dJ9/SE];J8:=[dJ8/S]; J7:=[dJ7/SW]; I7:=[dI7/W]; H7:=[dH7/NW]
I9	H9;IA;J9;I8	H9:=[dH9/N]; IA:=[dIA/E]; J9:=[dJ9/S]; I8:=[dI8/W]
IA	HA;IB;JA;I9	HA:=[dHA/N]; IB:=[dIB/E]; JA:=[dJA/S]; I9:=[dI9/W]
IB	HB;IC;JB;IA	HB:=[dHB/N]; IC:=[dIC/E]; JB:=[dJB/S]; IA:=[dIA/W]
IC	HC;ID;JC;IB	HC:=[dHC/N]; ID:=[dID/E]; JC:=[dJC/S]; IB:=[dIB/W]
ID	HD;IE;JD;IC	HD:=[dHD/N]; IE:=[dIE/E]; JD:=[dJD/S]; IC:=[dIC/W]
IE	HE;IF;JE;ID	HE:=[dHE/N]; IF:=[dIF/E]; JE:=[dJE/S]; ID:=[dID/W]
IF	HF;IG;JF;IE	HF:=[dHF/N]; IG:=[dIG/E]; JF:=[dJF/S]; IE:=[dIE/W]
IG	HG;JG;IF	HG:=[dHG/N]; JG:=[dJG/S]; IF:=[dIF/W]

J0	I0;J1;K1;K0	I0:=[dI0/N]; J1:=[dJ1/E]; K1:=[dK1/SE]; K0:=[dK0/S]
J1	I1;J2;K1;J0	I1:=[dI1/N]; J2:=[dJ2/E]; K1:=[dK1/S]; J0:=[dJ0/W]
J2	I2;J3;K2;J1	I2:=[dI2/N]; J3:=[dJ3/E]; K2:=[dK2/S]; J1:=[dJ1/W]
J3	I3;J4;K3;J2	I3:=[dI3/N]; J4:=[dJ4/E]; K3:=[dK3/S]; J2:=[dJ2/W]
J4	I4;J5;K4;J3	I4:=[dI4/N]; J5:=[dJ5/E]; K4:=[dK4/S]; J3:=[dJ3/W]
J5	I5;J6;K5;J4	I5:=[dI5/N]; J6:=[dJ6/E]; K5:=[dK5/S]; J4:=[dJ4/W]
J6	I6;J7;K6;J5	I6:=[dI6/N]; J7:=[dJ7/E]; K6:=[dK6/S]; J5:=[dJ5/W]
J7	I7;I8;J8;K7;K6;J6	I7:=[dI7/N]; I8:=[dI8/NE]; J8:=[dJ8/E]; K7:=[dK7/S]; K6:=[dK6/SW]; J6:=[dJ6/W]
J8	I8;J9;K8;J7	I8:=[dI8/N]; J9:=[dJ9/E]; K8:=[dK8/S]; J7:=[dJ7/W]
J9	I9;JA;KA;K9;J8;I8	I9:=[dI9/N]; JA:=[dJA/E]; KA:=[dKA/SE]; K9:=[dK9/S]; J8:=[dJ8/W]; I8:=[dI8/NW]
JA	IA;JB;KA;J9	IA:=[dIA/N]; JB:=[dJB/E]; KA:=[dKA/S]; J9:=[dJ9/W]
JB	IB;JC;KB;JA	IB:=[dIB/N]; JC:=[dJC/E]; KB:=[dKB/S]; JA:=[dJA/W]
JC	IC;JD;KC;JB	IC:=[dIC/N]; JD:=[dJD/E]; KC:=[dKC/S]; JB:=[dJB/W]
JD	ID;JE;KD;JC	ID:=[dID/N]; JE:=[dJE/E]; KD:=[dKD/S]; JC:=[dJC/W]
JE	IE;JF;KE;JD	IE:=[dIE/N]; JF:=[dJF/E]; KE:=[dKE/S]; JD:=[dJD/W]
JF	IF;JG;KF;JE	IF:=[dIF/N]; JG:=[dJG/E]; KF:=[dKF/S]; JE:=[dJE/W]
JG	IG;KG;KF;JF	IG:=[dIG/N]; KG:=[dKG/S]; KF:=[dKF/SW]; JF:=[dJF/W]
K0	J0;K1;L0	J0:=[dJ0/N]; K1:=[dK1/E]; L0:=[dL0/S]
K1	J1;K2;L2;L1;K0;J0	J1:=[dJ1/N]; K2:=[dK2/E]; L2:=[dL2/SE]; L1:=[dL1/S]; K0:=[dK0/W]; J0:=[dJ0/NW]
K2	J2;K3;L2;K1	J2:=[dJ2/N]; K3:=[dK3/E]; L2:=[dL2/S]; K1:=[dK1/W]
K3	J3;K4;L3;K2	J3:=[dJ3/N]; K4:=[dK4/E]; L3:=[dL3/S]; K2:=[dK2/W]
K4	J4;K5;L4;K3	J4:=[dJ4/N]; K5:=[dK5/E]; L4:=[dL4/S]; K3:=[dK3/W]
K5	J5;K6;L5;K4	J5:=[dJ5/N]; K6:=[dK6/E]; L5:=[dL5/S]; K4:=[dK4/W]
K6	J6;J7;K7;L6;L5;K5	J6:=[dJ6/N]; J7:=[dJ7/NE]; K7:=[dK7/E]; L6:=[dL6/S]; L5:=[dL5/SW]; K5:=[dK5/W]
K7	J7;K8;L7;K6	J7:=[dJ7/N]; K8:=[dK8/E]; L7:=[dL7/S]; K6:=[dK6/W]
K8	J8;K9;L8;K7	J8:=[dJ8/N]; K9:=[dK9/E]; L8:=[dL8/S]; K7:=[dK7/W]
K9	J9;KA;L9;K8	J9:=[dJ9/N]; KA:=[dKA/E]; L9:=[dL9/S]; K8:=[dK8/W]
KA	JA;KB;LB;LA;K9;J9	JA:=[dJA/N]; KB:=[dKB/E]; LB:=[dLB/SE]; LA:=[dLA/S]; K9:=[dK9/W]; J9:=[dJ9/NW]
KB	JB;KC;LB;KA	JB:=[dJB/N]; KC:=[dKC/E]; LB:=[dLB/S]; KA:=[dKA/W]
KC	JC;KD;LC;KB	JC:=[dJC/N]; KD:=[dKD/E]; LC:=[dLC/S]; KB:=[dKB/W]
KD	JD;KE;LD;KC	JD:=[dJD/N]; KE:=[dKE/E]; LD:=[dLD/S]; KC:=[dKC/W]
KE	JE;KF;LE;KD	JE:=[dJE/N]; KF:=[dKF/E]; LE:=[dLE/S]; KD:=[dKD/W]
KF	JF;JG;KG;LF;LE;KE	JF:=[dJF/N]; JG:=[dJG/NE]; KG:=[dKG/E]; LF:=[dLF/S]; LE:=[dLE/SW]; KE:=[dKE/W]
KG	JG;LG;KF	JG:=[dJG/N]; LG:=[dLG/S]; KF:=[dKF/W]
L0	K0;L1;M0	K0:=[dK0/N]; L1:=[dL1/E]; M0:=[dM0/S]
L1	K1;L2;M1;L0	K1:=[dK1/N]; L2:=[dL2/E]; M1:=[dM1/S]; L0:=[dL0/W]
L2	K2;L3;M3;M2;L1;K1	K2:=[dK2/N]; L3:=[dL3/E]; M3:=[dM3/SE]; M2:=[dM2/S]; L1:=[dL1/W]; K1:=[dK1/NW]
L3	K3;L4;M3;L2	K3:=[dK3/N]; L4:=[dL4/E]; M3:=[dM3/S]; L2:=[dL2/W]
L4	K4;L5;M4;L3	K4:=[dK4/N]; L5:=[dL5/E]; M4:=[dM4/S]; L3:=[dL3/W]
L5	K5;K6;L6;M5;M4;L4	K5:=[dK5/N]; K6:=[dK6/NE]; L6:=[dL6/E]; M5:=[dM5/S]; M4:=[dM4/SW]; L4:=[dL4/W]
L6	K6;L7;M6;L5	K6:=[dK6/N]; L7:=[dL7/E]; M6:=[dM6/S]; L5:=[dL5/W]
L7	K7;L8;M7;L6	K7:=[dK7/N]; L8:=[dL8/E]; M7:=[dM7/S]; L6:=[dL6/W]
L8	K8;L9;M8;L7	K8:=[dK8/N]; L9:=[dL9/E]; M8:=[dM8/S]; L7:=[dL7/W]
L9	K9;LA;M9;L8	K9:=[dK9/N]; LA:=[dLA/E]; M9:=[dM9/S]; L8:=[dL8/W]
LA	KA;LB;MA;L9	KA:=[dKA/N]; LB:=[dLB/E]; MA:=[dMA/S]; L9:=[dL9/W]
LB	KB;LC;MC;MB;LA;KA	KB:=[dKB/N]; LC:=[dLC/E]; MC:=[dMC/SE]; MB:=[dMB/S]; LA:=[dLA/W]; KA:=[dKA/NW]
LC	KC;LD;MC;LB	KC:=[dKC/N]; LD:=[dLD/E]; MC:=[dMC/S]; LB:=[dLB/W]
LD	KD;LE;MD;LC	KD:=[dKD/N]; LE:=[dLE/E]; MD:=[dMD/S]; LC:=[dLC/W]
LE	KE;KF;LF;ME;MD;LD	KE:=[dKE/N]; KF:=[dKF/NE]; LF:=[dLF/E]; ME:=[dME/S]; MD:=[dMD/SW]; LD:=[dLD/W]
LF	KF;LG;MF;LE	LF:=[dLF/N]; LG:=[dLG/E]; MF:=[dMF/S]; LE:=[dLE/W]
LG	KG;MG;LF	KG:=[dKG/N]; MG:=[dMG/S]; LF:=[dLF/W]

henryvpham@gmail.com

M0	L0;M1;N0	L0:=[dL0/N]; M1:=[dM1/E]; N0:=[dN0/S]
M1	L1;M2;N1;M0	L1:=[dL1/N]; M2:=[dM2/E]; N1:=[dN1/S]; M0:=[dM0/W]
M2	L2;M3;N2;M1	L2:=[dL2/N]; M3:=[dM3/E]; N2:=[dN2/S]; M1:=[dM1/W]
M3	L3;M4;N4;N3;M2;L2	L3:=[dL3/N]; M4:=[dM4/E]; N4:=[dN4/SE]; N3:=[dN3/S]; M2:=[dM2/W]; L2:=[dL2/NW]
M4	L4;L5;M5;N4;N3;M3	L4:=[dL4/N]; L5:=[dL5/NE]; M5:=[dM5/E]; N4:=[dN4/S]; N3:=[dN3/SW]; M3:=[dM3/W]
M5	L5;M6;N5;M4	L5:=[dL5/N]; M6:=[dM6/E]; N5:=[dN5/S]; M4:=[dM4/W]
M6	L6;M7;N6;M5	L6:=[dL6/N]; M7:=[dM7/E]; N6:=[dN6/S]; M5:=[dM5/W]
M7	L7;M8;N7;M6	L7:=[dL7/N]; M8:=[dM8/E]; N7:=[dN7/S]; M6:=[dM6/W]
M8	L8;M9;N8;M7	L8:=[dL8/N]; M9:=[dM9/E]; N8:=[dN8/S]; M7:=[dM7/W]
M9	L9;MA;N9;M8	L9:=[dL9/N]; MA:=[dMA/E]; N9:=[dN9/S]; M8:=[dM8/W]
MA	LA;MB;NA;M9	LA:=[dLA/N]; MB:=[dMB/E]; NA:=[dNA/S]; M9:=[dM9/W]
MB	LB;MC;NB;MA	LB:=[dLB/N]; MC:=[dMC/E]; NB:=[dNB/S]; MA:=[dMA/W]
MC	LC;MD;ND;NC;MB;LB	LC:=[dLC/N]; MD:=[dMD/E]; ND:=[dND/SE]; NC:=[dNC/S]; MB:=[dMB/W]; LB:=[dLB/NW]
MD	LD;LE;ME;ND;NC;MC	LD:=[dLD/N]; LE:=[dLE/NE]; ME:=[dME/E]; ND:=[dND/S]; NC:=[dNC/SW]; MC:=[dMC/W]
ME	LE;MF;NE;MD	LE:=[dLE/N]; MF:=[dMF/E]; NE:=[dNE/S]; MD:=[dMD/W]
MF	LF;MG;NF;ME	LF:=[dLF/N]; MG:=[dMG/E]; NF:=[dNF/S]; ME:=[dME/W]
MG	LG;NG;MF	LG:=[dLG/N]; NG:=[dNG/S]; MF:=[dMF/W]
N0	M0;N1;O0	M0:=[dM0/N]; N1:=[dN1/E]; O0:=[dO0/S]
N1	M1;N2;O1;N0	M1:=[dM1/N]; N2:=[dN2/E]; O1:=[dO1/S]; N0:=[dN0/W]
N2	M2;N3;O2;N1	M2:=[dM2/N]; N3:=[dN3/E]; O2:=[dO2/S]; N1:=[dN1/W]
N3	M3;M4;N4;O3;O2;N2	M3:=[dM3/N]; M4:=[dM4/NE]; N4:=[dN4/E]; O3:=[dO3/S]; O2:=[dO2/SW]; N2:=[dN2/W]
N4	M4;N5;O5;O4;N3;M3	M4:=[dM4/N]; N5:=[dN5/E]; O5:=[dO5/SE]; O4:=[dO4/S]; N3:=[dN3/W]; M3:=[dM3/NW]
N5	M5;N6;O5;N4	M5:=[dM5/N]; N6:=[dN6/E]; O5:=[dO5/S]; N4:=[dN4/W]
N6	M6;N7;O6;N5	M6:=[dM6/N]; N7:=[dN7/E]; O6:=[dO6/S]; N5:=[dN5/W]
N7	M7;N8;O7;N6	M7:=[dM7/N]; N8:=[dN8/E]; O7:=[dO7/S]; N6:=[dN6/W]
N8	M8;N9;O8;N7	M8:=[dM8/N]; N9:=[dN9/E]; O8:=[dO8/S]; N7:=[dN7/W]
N9	M9;NA;O9;N8	M9:=[dM9/N]; NA:=[dNA/E]; O9:=[dO9/S]; N8:=[dN8/W]
NA	MA;NB;OA;N9	MA:=[dMA/N]; NB:=[dNB/E]; OA:=[dOA/S]; N9:=[dN9/W]
NB	MB;NC;OB;NA	MB:=[dMB/N]; NC:=[dNC/E]; OB:=[dOB/S]; NA:=[dNA/W]
NC	MC;MD;ND;OC;OB;NB	MC:=[dMC/N]; MD:=[dMD/NE]; ND:=[dND/E]; OC:=[dOC/S]; OB:=[dOB/SW]; NB:=[dNB/W]
ND	MD;NE;OE;OD;NC;MC	MD:=[dMD/N]; NE:=[dNE/E]; OE:=[dOE/SE]; OD:=[dOD/S]; NC:=[dNC/W]; MC:=[dMC/NW]
NE	ME;NF;OE;ND	ME:=[dME/N]; NF:=[dNF/E]; OE:=[dOE/S]; ND:=[dND/W]
NF	MF;NG;OF;NE	MF:=[dMF/N]; NG:=[dNG/E]; OF:=[dOF/S]; NE:=[dNE/W]
NG	MG;OG;NF	MG:=[dMG/N]; OG:=[dOG/S]; NF:=[dNF/W]
O0	N0;O1;P0	N0:=[dN0/N]; O1:=[dO1/E]; P0:=[dP0/S]
O1	N1;O2;P1;O0	N1:=[dN1/N]; O2:=[dO2/E]; P1:=[dP1/S]; O0:=[dO0/W]
O2	N2;N3;O3;P2;P1;O1	N2:=[dN2/N]; N3:=[dN3/NE]; O3:=[dO3/E]; P2:=[dP2/S]; P1:=[dP1/SW]; O1:=[dO1/W]
O3	N3;O4;P3;O2	N3:=[dN3/N]; O4:=[dO4/E]; P3:=[dP3/S]; O2:=[dO2/W]
O4	N4;O5;P4;O3	N4:=[dN4/N]; O5:=[dO5/E]; P4:=[dP4/S]; O3:=[dO3/W]
O5	N5;O6;P6;P5;O4;N4	N5:=[dN5/N]; O6:=[dO6/E]; P6:=[dP6/SE]; P5:=[dP5/S]; O4:=[dO4/W]; N4:=[dN4/NW]
O6	N6;O7;P6;O5	N6:=[dN6/N]; O7:=[dO7/E]; P6:=[dP6/S]; O5:=[dO5/W]
O7	N7;O8;P7;O6	N7:=[dN7/N]; O8:=[dO8/E]; P7:=[dP7/S]; O6:=[dO6/W]
O8	N8;O9;P8;O7	N8:=[dN8/N]; O9:=[dO9/E]; P8:=[dP8/S]; O7:=[dO7/W]
O9	N9;OA;P9;O8	N9:=[dN9/N]; OA:=[dOA/E]; P9:=[dP9/S]; O8:=[dO8/W]
OA	NA;OB;PA;O9	NA:=[dNA/N]; OB:=[dOB/E]; PA:=[dPA/S]; O9:=[dO9/W]
OB	NB;NC;OC;PB;PA;OA	NB:=[dNB/N]; NC:=[dNC/NE]; OC:=[dOC/E]; PB:=[dPB/S]; PA:=[dPA/SW]; OA:=[dOA/W]
OC	NC;OD;PC;OB	NC:=[dNC/N]; OD:=[dOD/E]; PC:=[dPC/S]; OB:=[dOB/W]
OD	ND;OE;PD;OC	ND:=[dND/N]; OE:=[dOE/E]; PD:=[dPD/S]; OC:=[dOC/W]
OE	NE;OF;PF;PE;OD;ND	NE:=[dNE/N]; OF:=[dOF/E]; PF:=[dPF/SE]; PE:=[dPE/S]; OD:=[dOD/W]; ND:=[dND/NW]
OF	NF;OG;PF;OE	NF:=[dNF/N]; OG:=[dOG/E]; PF:=[dPF/S]; OE:=[dOE/W]
OG	NG;PG;OF	NG:=[dNG/N]; PG:=[dPG/S]; OF:=[dOF/W]

P0	O0;P1;Q0	O0:=[dO0/N]; P1:=[dP1/E]; Q0:=[dQ0/S]
P1	O1;O2;P2;Q1;Q0;P0	O1:=[dO1/N]; O2:=[dO2/NE]; P2:=[dP2/E]; Q1:=[dQ1/S]; Q0:=[dQ0/SW]; P0:=[dP0/W]
P2	O2;P3;Q2;P1	O2:=[dO2/N]; P3:=[dP3/E]; Q2:=[dQ2/S]; P1:=[dP1/W]
P3	O3;P4;Q3;P2	O3:=[dO3/N]; P4:=[dP4/E]; Q3:=[dQ3/S]; P2:=[dP2/W]
P4	O4;P5;Q4;P3	O4:=[dO4/N]; P5:=[dP5/E]; Q4:=[dQ4/S]; P3:=[dP3/W]
P5	O5;P6;Q5;P4	O5:=[dO5/N]; P6:=[dP6/E]; Q5:=[dQ5/S]; P4:=[dP4/W]
P6	O6;P7;Q7;Q6;P5;O5	O6:=[dO6/N]; P7:=[dP7/E]; Q7:=[dQ7/SE]; Q6:=[dQ6/S]; P5:=[dP5/W]; O5:=[dO5/NW]
P7	O7;P8;Q7;P6	O7:=[dO7/N]; P8:=[dP8/E]; Q7:=[dQ7/S]; P6:=[dP6/W]
P8	O8;P9;Q8;P7	O8:=[dO8/N]; P9:=[dP9/E]; Q8:=[dQ8/S]; P7:=[dP7/W]
P9	O9;PA;Q9;P8	O9:=[dO9/N]; PA:=[dPA/E]; Q9:=[dQ9/S]; P8:=[dP8/W]
PA	OA;OB;PB;QA;Q9;P9	OA:=[dOA/N]; OB:=[dOB/NE]; PB:=[dPB/E]; QA:=[dQA/S]; Q9:=[dQ9/SW]; P9:=[dP9/W]
PB	OB;PC;QB;PA	OB:=[dOB/N]; PC:=[dPC/E]; QB:=[dQB/S]; PA:=[dPA/W]
PC	OC;PD;QC;PB	OC:=[dOC/N]; PD:=[dPD/E]; QC:=[dQC/S]; PB:=[dPB/W]
PD	OD;PE;QD;PC	OD:=[dOD/N]; PE:=[dPE/E]; QD:=[dQD/S]; PC:=[dPC/W]
PE	OE;PF;QE;PD	OE:=[dOE/N]; PF:=[dPF/E]; QE:=[dQE/S]; PD:=[dPD/W]
PF	OF;PG;QG;QF;PE;OE	OF:=[dOF/N]; PG:=[dPG/E]; QG:=[dQG/SE]; QF:=[dQF/S]; PE:=[dPE/W]; OE:=[dOE/NW]
PG	OG;QG;PF	OG:=[dOG/N]; QG:=[dQG/S]; PF:=[dPF/W]
Q0	P0;P1;Q1	P0:=[dP0/N]; P1:=[dP1/NE]; Q1:=[dB0/E]
Q1	P1;Q2;Q0	P1:=[dP1/N]; Q2:=[dQ2/E]; Q0:=[dQ0/W]
Q2	P2;Q3;Q1	P2:=[dP2/N]; Q3:=[dQ3/E]; Q1:=[dQ1/W]
Q3	P3;Q4;Q2	P3:=[dP3/N]; Q4:=[dQ4/E]; Q2:=[dQ2/W]
Q4	P4;Q5;Q3	P4:=[dP4/N]; Q5:=[dQ5/E]; Q3:=[dQ3/W]
Q5	P5;Q6;Q4	P5:=[dP5/N]; Q6:=[dQ6/E]; Q4:=[dQ4/W]
Q6	P6;Q7;Q5	P6:=[dP6/N]; Q7:=[dQ7/E]; Q5:=[dQ5/W]
Q7	P7;Q8;Q6;P6	P7:=[dP7/N]; Q8:=[dQ8/E]; Q6:=[dQ6/W]; P6:=[dP6/NW]
Q8	P8;Q9;Q7	P8:=[dP8/N]; Q9:=[dQ9/E]; Q7:=[dQ7/W]
Q9	P9;PA;QA;Q8	P9:=[dP9/N]; PA:=[dPA/NE]; QA:=[dQA/E]; Q8:=[dQ8/W]
QA	PA;QB;Q9	PA:=[dPA/N]; QB:=[dQB/E]; Q9:=[dQ9/W]
QB	PB;QC;QA	PB:=[dPB/N]; QC:=[dQC/E]; QA:=[dQA/W]
QC	PC;QD;QB	PC:=[dPC/N]; QD:=[dQD/E]; QB:=[dQB/W]
QD	PD;QE;QC	PD:=[dPD/N]; QE:=[dQE/E]; QC:=[dQC/W]
QE	PE;QF;QD	PE:=[dPE/N]; QF:=[dQF/E]; QD:=[dQD/W]
QF	PF;QG;QE	PF:=[dPF/N]; QG:=[dQG/E]; QE:=[dQE/W]
QG	PG;QF;PF	PG:=[dPG/N]; QF:=[dQF/W]; PF:=[dPF/NW]
SQUARE MATRIX TABLE NODES' PROPERTY		

Table-1: NEIGHBOR-TO-NEIGHBORS NODES' PROPERTIES TABLE

Standard G-ROUTING ALGORITHM's Rules

The G-ROUTING ALGORITHM has the following defined rules.

1. Network Database must define each node with its neighbor nodes' properties for Neighbor-to-Neighbors networking methodology with directions and distances from a node to its neighbor nodes.
2. Always start at 12 O' clock in clockwise direction when looking for node's neighbors.
3. Always try clockwise direction when reaching disconnected node, then back to the original direction when possible.

4. Iterate through neighbor nodes for both source and destination nodes to find sharing node and eliminate the duplicated nodes from previous steps. This means that the iterations will start from the source node iterate to the destination node and from destination node iterate to the source node until find a sharing node.
5. When found a shared node from both source and destination nodes' paths, always look for the closest node that contains this shared node and previous nodes that link to this shared node from both source and destination directions.
6. Always try optimization routing with the network layouts in uniform geometry like all squares layout or all hexagons layout for faster routing; optimization searching four outward directions in 90° for square layout and six outward directions in 30° for hexagon layout.
7. For the non-uniform geometry network layouts, after try optimization routing to find the crossing section, then apply the standard G-ROUTING ALGORITHM method in clockwise direction for both source and destination nodes within the crossed section toward each other direction in degree angles within the crossed section. This way the algorithm can find the shortest or shortcut routes to the destination node.

Standard G-ROUTING ALGORITHM Methodology

Table-2 below shows the standard G-ROUTING ALGORITHM for **Figure-1** above in uniform network layout with all squares in geometry layout start with the source node '**C5**' to find the destination node '**MC**'. Below are the sequences to start searching for routes with the standard G-ROUTING ALGORITHM.

1. Start with source node '**C5**' on the **Left-Side** and based on the database **Table-1**, there are 4 neighbor nodes '**B5, C6, D5, C4**' in order of clockwise direction start at 12 O' clock; the following step '1.1' starts with 1st node '**B5**' which has 3 neighbor nodes '**A5, B6, B4**' not including the source node 'C5'; the next step '1.2' starts with 2nd node '**C6**' which has 3 neighbor nodes '**B6**, **C7, D6**' not including the source node 'C5' but found duplicated node '**B6**' from the previous step '1.1' and this node '**B6**' is shown in **purple** and should **be eliminated** from the list in computer programming algorithm; the next steps would be the same for the other nodes '**D5**' and '**C4**' in steps '1.3' and '1.4'.

Pham, Henry V.
henryvpham@gmail.com

2. Start with destination node '**MC'** on the **Right-Side**, there are 4 neighbor nodes '**LC, MD, NC, MB**' in order of clockwise direction start at 12 O' clock; the following step '1.1' starts with 1st node '**LC**' which has 3 neighbor nodes '**KC, LD, LB**' not including the original destination node 'MC'; the next step '1.2' starts with 2nd node '**MD**' which has 3 neighbor nodes '**LD, ME, ND**' not including the original destination node 'MC' but found duplicated node '**LD**' from the previous step '1.1' and this node '**LD**' is shown in **purple** and should **be eliminated** from the list in computer programming algorithm; the next steps would be the same for the other nodes '**NC**' and '**MB**' in steps '1.3' and '1.4'.
3. Back to the **Left-Side** in sequence 1) above, starts with step '1.1.1' for 1st node '**A5**' from previous step '1.1', and there are 3 neighbor nodes '**A6, B5, A4**' with duplicated node '**B5**' from previous step; the following step '1.1.2' for 2nd node '**B6**' from previous step '1.1', and there are 4 neighbor nodes '**A6, B7, C6, B5**' with duplicated nodes '**A6, C6, B5**' from previous steps; the next steps would be the same for other nodes '**B4**' in step '1.1.3' from its previous step '1.1', and node '**C7**' in step '1.2.2' from its previous step '1.2' for the 2nd node.
4. Back to the **Right-Side** in sequence 2) above, starts with step '1.1.1' for 1st node '**KC**' from previous step '1.1', and there are 4 neighbor nodes '**JC, KD, LC, KB**' with duplicated node '**LC**' from previous step; the following step '1.1.2' for 2nd node '**LD**' from previous step '1.1', and there are 4 neighbor nodes '**KD, LE, MD, LC'** with duplicated nodes '**KD, MD, LC**' from previous steps; the next steps would be the same for other nodes '**LB**' in step '1.1.3' from previous step '1.1', and node '**ME**' in step '1.2.2' from previous step '1.2' for the 2nd node.
5. Repeat the above sequences for both right and Left-Sides for both source and destination nodes until found a shared node '**EC**' which is highlight in **green** as shown in step '1.1.2.2.2.2.2.2.3' of node '**DC**' on **Left-Side** and in step '1.1.1.1.1.1.1.1' of node '**FC**' on **Right-Side**. This node '**EC**' will be connected to the 1st route for this algorithm. With 2 more iterations as shown at the end of **Table-2**, the algorithm will show 2 more shared nodes '**FB**' and '**GA**' for 2 more routes.
6. Now, links the 1st shared node '**EC**' to find the 1st route. Starts from the **Left-Side**, node '**EC**' is connected to '**DC**', node '**DC**' is connected to node '**CC**', node '**CC**' is connected to '**CB**', node '**CB**' is connected to '**CA**', node '**CA**' is connected to '**C9**',

node '**C9**' is connected to '**C8**', node '**C8**' is connected to '**C7**', node '**C7**' is connected to '**C6**', and node '**C6**' is connected to the source node '**C5**'. Then starts from the **Right-Side**, node '**EC**' is connected to '**FC**', node '**FC**' is connected to '**GC**', node '**GC**' is connected to '**HC**', node '**HC**' is connected to '**IC**', node '**IC**' is connected to '**JC**', node '**JC**' is connected '**KC**', node '**KC**' is connected to '**LC**', and node '**LC**' is connected to the destination node '**MC**'. So, the 1st route would be in the following series of nodes '**C5**, **C6, C7, C8, C9, CA, CB, CC, DC, EC, FC, GC, HC, IC, JC, KC, LC, MC**'.

7. With 2 more iterations as shown at the end of **Table-2**, the 2nd shared node is '**FB**' and will create the 2nd route would be in the following series of nodes '**C5**, **C6, C7, C8, C9, CA, CB, DB, EB, FB, FC, GC, HC, IC, JC, KC, LC, MC**'; this would be the 2nd route to the destination node '**MC**'. The last row from Table-2 shows the 3rd shared node is '**GA**' and will create the 3rd route would be in the following series of nodes '**C5**, **C6, C7, C8, C9, CA, DA, EA, FA, GA, GB, GC, HC, IC, JC, KC, LC, MC**'; this would be the 3rd route to the destination node '**MC**'.
8. Now links to the database for all nodes' properties in **Table-1**, **Table-3** below shows the complete routing tables for all 3 routes with the routing paths in series of nodes with total distances of each route and how many turns each route takes. **Table-3** below shows the distance of each node to the previous node in a general form of "d<Node>" stands for pre-calculated distance 'd' of a node which is defined in the system database. The 1st route with total of 18 nodes includes the source and destination nodes with total of 1 turn from East to South direction. The 2nd route also with total of 18 nodes but with total of 3 turns from East to South to East, then to South directions. The 3rd route also with total of 18 nodes but with total of 3 turns from East to South to East, then to South directions. Calculate total distance of each route to find out the shortest route; and with number of turns. So, the 1st route is the best route found from source node '**C5**' to the destination node '**MC**'.
9. Finally, **Figure-3** shows the graphical directions for all 3 routes above. The directions in Figure-3 with denoted directions of compact symbol with North direction on top. The algorithm shows the 1st route as the fastest route which makes perfectly matching with expectation in geometry layout. Note that computer programming algorithm will only based on the node's properties database as shown in **Table-1**

above to start with the routing nodes from the source node and the destination node step-by-step as shown in sequence '1.' to '5.' above.

START SOURCE NODE 'C5'				START DESTINATION NODE 'MC'		
Steps	Node	Neighbor Nodes		Steps	Node	Neighbor Nodes
1	C5	B5, C6, D5, C4		1	MC	LC, MD, NC, MB
1.1	B5	A5, B6, B4		1.1	LC	KC, LD, LB
1.2	C6	B6, C7, D6		1.2	MD	LD, ME, ND
1.3	D5	D6, E5, D4		1.3	NC	ND, OC, NB
1.4	C4	B4, D4, C3		1.4	MB	LB, NB, MA
1.1.1	A5	A6, B5, A4		1.1.1	KC	JC, KD, LC, KB
1.1.2	B6	A6, B7, C6, B5		1.1.2	LD	KD, LE, MD, LC
1.1.3	B4	A4, B5, C4, B3		1.1.3	LB	KB, LC, MB, LA
1.2.2	C7	B7, C8, D7, C6		1.2.2	ME	LE, MF, NE, MD
1.2.3	D6	C6, D7, E6, D5		1.2.3	ND	MD, NE, OD, NC
1.3.2	E5	D5, E6, F5, E4		1.3.2	OC	NC, OD, PC, OB
1.3.3	D4	C4, D5, E4, D3		1.3.3	NB	MB, NC, OB, NA
1.4.3	C3	B3, C4, D3, C2		1.4.3	MA	LA, MB, NA, M9
1.1.1.1	A6	A7, B6, A5		1.1.1.1	JC	IC, JD, KC, JB
1.1.1.3	A4	A5, B4, A3		1.1.1.2	KD	JD, KE, LD, KC
1.1.2.2	B7	A7, B8, C7, B6		1.1.1.4	KB	JB, KC, LB, KA
1.1.3.4	B3	A3, B4, C3, B2		1.1.2.2	LE	KE, LF, ME, LD
1.2.2.2	C8	B8, C9, D8, C7		1.1.3.4	LA	KA, LB, MA, L9
1.2.2.3	D7	C7, D8, E7, D6		1.2.2.2	MF	LF, MG, NF, ME
1.2.3.3	E6	D6, E7, F6, E5		1.2.2.3	NE	ME, NF, OE, ND
1.3.2.3	F5	E5, F6, G5, F4		1.2.3.3	OD	ND, OE, PD, OC
1.3.2.4	E4	D4, E5, F4, E3		1.3.2.3	PC	OC, PD, QC, PB
1.3.3.4	D3	C3, D4, E3, D2		1.3.2.4	OB	NB, OC, PB, OA
1.4.3.4	C2	B2, C3, D2, C1		1.3.3.4	NA	MA, NB, OA, N9
1.1.1.1.1	A7	A8, B7, A6		1.4.3.4	M9	L9, MA, N9, M8
1.1.1.3.3	A3	A4, B3, A2		1.1.1.1.1	IC	HC, ID, JC, IB
1.1.2.2.2	B8	A8, B9, C8, B7		1.1.1.1.2	JD	ID, JE, KD, JC
1.1.3.4.4	B2	A2, B3, C2, B1		1.1.1.1.4	JB	IB, JC, KB, JA
1.2.2.2.2	C9	B9, CA, D9, C8		1.1.1.2.2	KE	JE, KF, LE, KD
1.2.2.2.3	D8	C8, D9, E8, D7		1.1.1.4.4	KA	JA, KB, LA, K9
1.2.2.3.3	E7	D7, E8, F7, E6		1.1.2.2.2	LF	KF, LG, MF, LE
1.2.3.3.3	F6	E6, F7, G6, F5		1.1.3.4.4	L9	K9, LA, M9, L8
1.3.2.3.3	G5	F5, G6, H5, G4		1.2.2.2.2	MG	LG, NG, MF
1.3.2.3.4	F4	E4, F5, G4, F3		1.2.2.2.3	NF	MF, NG, OF, NE
1.3.2.4.4	E3	D3, E4, F3, E2		1.2.2.3.3	OE	NE, OF, PE, OD
1.3.3.4.4	D2	C2, D3, E2, D1		1.2.3.3.3	PD	OD, PE, QD, PC
1.4.3.4.4	C1	B1, C2, D1, C0		1.3.2.3.3	QC	PC, QD, QB
1.1.1.1.1.1	A8	A9, B8, A7		1.3.2.3.4	PB	OB, PC, QB, PA
1.1.1.3.3.3	A2	A3, B2, A1		1.3.2.4.4	OA	NA, OB, PA, O9
1.1.2.2.2.2	B9	A9, BA, C9, B8		1.3.3.4.4	N9	M9, NA, O9, N8
1.1.3.4.4.4	B1	A1, B2, C1, B0		1.4.3.4.4	M8	L8, M9, N8, M7
1.2.2.2.2.2	CA	BA, CB, DA, C9		1.1.1.1.1.1	HC	GC, HD, IC, HB
1.2.2.2.2.3	D9	C9, DA, E9, D8		1.1.1.1.1.2	ID	HD, IE, JD, IC
1.2.2.2.3.3	E8	D8, E9, F8, E7		1.1.1.1.1.4	IB	HB, IC, JB, IA

1.2.2.3.3.3	F7	E7, F8, G7, F6		1.1.1.1.2.2	JE	IE, JF, KE, JD
1.2.3.3.3.3	G6	F6, G7, H6, G5		1.1.1.1.4.4	JA	IA, JB, KA, J9
1.3.2.3.3.3	H5	G5, H6, I5, H4		1.1.1.2.2.2	KF	JF, KG, LF, KE
1.3.2.3.3.4	G4	F4, G5, H4, G3		1.1.1.4.4.4	K9	J9, KA, L9, K8
1.3.2.3.4.4	F3	E3, F4, G3, F2		1.1.2.2.2.2	LG	KG, MG, LF
1.3.2.4.4.4	E2	D2, E3, F2, E1		1.1.3.4.4.4	L8	K8, L9, M8, L7
1.3.3.4.4.4	D1	C1, D2, E1, D0		1.2.2.2.2.2	NG	MG, OG, NF
1.4.3.4.4.4	C0	B0, C1, D0		1.2.2.2.3.3	OF	NF, OG, PF, OE
1.1.1.1.1.1.1	A9	AA, B9, A8		1.2.2.3.3.3	PE	OE, PF, QE, PD
1.1.1.3.3.3.3	A1	A2, B1, A0		1.2.3.3.3.3	QD	PD, QE, QC
1.1.2.2.2.2.2	BA	AA, BB, CA, B9		1.3.2.3.3.3	QB	PB, QC, QA
1.1.3.4.4.4.4	B0	A0, B1, C0		1.3.2.3.4.4	PA	OA, PB, QA, P9
1.2.2.2.2.2.2	CB	BB, CC, DB, CA		1.3.2.4.4.4	O9	N9, OA,P9, O8
1.2.2.2.2.2.3	DA	CA, DB, EA, D9		1.3.3.4.4.4	N8	M8, N9, O8, N7
1.2.2.2.2.3.3	E9	D9, EA, F9, E8		1.4.3.4.4.4	M7	L7, M8, N7, M6
1.2.2.2.3.3.3	F8	E8, F9, G8, F7		1.1.1.1.1.1.1	GC	FC, GD, HC, GB
1.2.2.3.3.3.3	G7	F7, G8, H7, G6		1.1.1.1.1.1.2	HD	GD, HE, ID, HC
1.2.3.3.3.3.3	H6	G6, H7, I6, H5		1.1.1.1.1.1.4	HB	GB, HC, IB, HA
1.3.2.3.3.3.3	I5	H5, I6, J5, I4		1.1.1.1.1.2.2	IE	HE, IF, JE, ID
1.3.2.3.3.3.4	H4	G4, H5, I4, H3		1.1.1.1.1.4.4	IA	HA, IB, JA, I9
1.3.2.3.3.4.4	G3	F3, G4, H3, G2		1.1.1.1.2.2.2	JF	IF, JG, KF, JE
1.3.2.3.4.4.4	F2	E2, F3, G2, F1		1.1.1.1.4.4.4	J9	I9, JA, K9, J8
1.3.2.4.4.4.4	E1	D1, E2, F1, E0		1.1.1.2.2.2.2	KG	JG, LG, KF
1.3.3.4.4.4.4	D0	C0, D1, E0		1.1.1.4.4.4.4	K8	J8, K9, L8, K7
1.1.1.1.1.1.1.1	AA	AB, BA, A9		1.1.3.4.4.4.4	L7	K7, L8, M7, L6
1.1.1.3.3.3.3.3	A0	A1, B0		1.2.2.2.2.2.2	OG	NG, PG, OF
1.1.2.2.2.2.2.2	CC	BC, CD, DC, CB		1.2.2.2.3.3.3	PF	OF, PG, QF, PE
1.1.2.2.2.2.2.3	DB	CB, DC, EB, DA		1.2.2.3.3.3.3	QE	PE, QF, QD
1.2.2.2.2.2.3.3	EA	DA, EB, FA, E9		1.3.2.3.4.4.4	P9	O9, PA, Q9, P8
1.2.2.2.2.3.3.3	F9	E9, FA, G9, F8		1.3.2.4.4.4.4	O8	N8, O9, P8, O7
1.2.2.2.3.3.3.3	G8	F8, G9, H8, G7		1.3.3.4.4.4.4	N7	M7, N8, O7, N6
1.2.2.3.3.3.3.3	H7	G7, H8, I7, H6		1.4.3.4.4.4.4	M6	L6, M7, N6, M5
1.2.3.3.3.3.3.3	I6	H6, I7, J6, I5		1.1.1.1.1.1.1.1	FC	EC, FD, GC, FB
1.3.2.3.3.3.3.3	J5	I5, J6, K5, J4		1.1.1.1.1.1.1.2	GD	FD, GE, HD, GC
1.3.2.3.3.3.3.4	I4	H4, I5, J4, I3		1.1.1.1.1.1.1.4	GB	FB, GC, HB, GA
1.3.2.3.3.3.4.4	H3	G3, H4, I3, H2		1.1.1.1.1.1.2.2	HE	GE, HF, IE, HD
1.3.2.3.3.4.4.4	G2	F2, G3, H2, G1		1.1.1.1.1.1.4.4	HA	GA, HB, IA, H9
1.3.2.3.4.4.4.4	F1	E1, F2, G1, F0		1.1.1.1.1.2.2.2	IF	HF, IG, JF, IE
1.3.2.4.4.4.4.4	E0	D0, E1, F0		1.1.1.1.1.4.4.4	I9	H9, IA, J9, I8
1.1.1.1.1.1.1.1.1	AB	AC, BB, AA		1.1.1.1.2.2.2.2	JG	IG, KG, JF
1.1.2.2.2.2.2.2.1	BC	AC, BD, CC, BB		1.1.1.1.4.4.4.4	J8	I8, J9, K8, J7
1.1.2.2.2.2.2.2.2	CD	BD, CE, DD, CC		1.1.1.4.4.4.4.4	K7	J7;K8;L7;K6
1.1.2.2.2.2.2.2.3	DC	CC, DD, EC, DB		1.1.3.4.4.4.4.4	L6	K6;L7;M6;L5
1.1.2.2.2.2.2.3.3	EB	DB, EC, FB, EA		1.2.2.2.2.2.2.2	PG	OG;QG;PF
1.2.2.2.2.2.3.3.3	FA	EA, FB, GA, F9		1.2.2.2.3.3.3.3	QF	PF;QG;QE
STANDARD G-ROUTING ALGORITHM (CLOCKWISE DIRECTIONS)						

Table-2: STANDARD G-ROUTING ALGORITHM

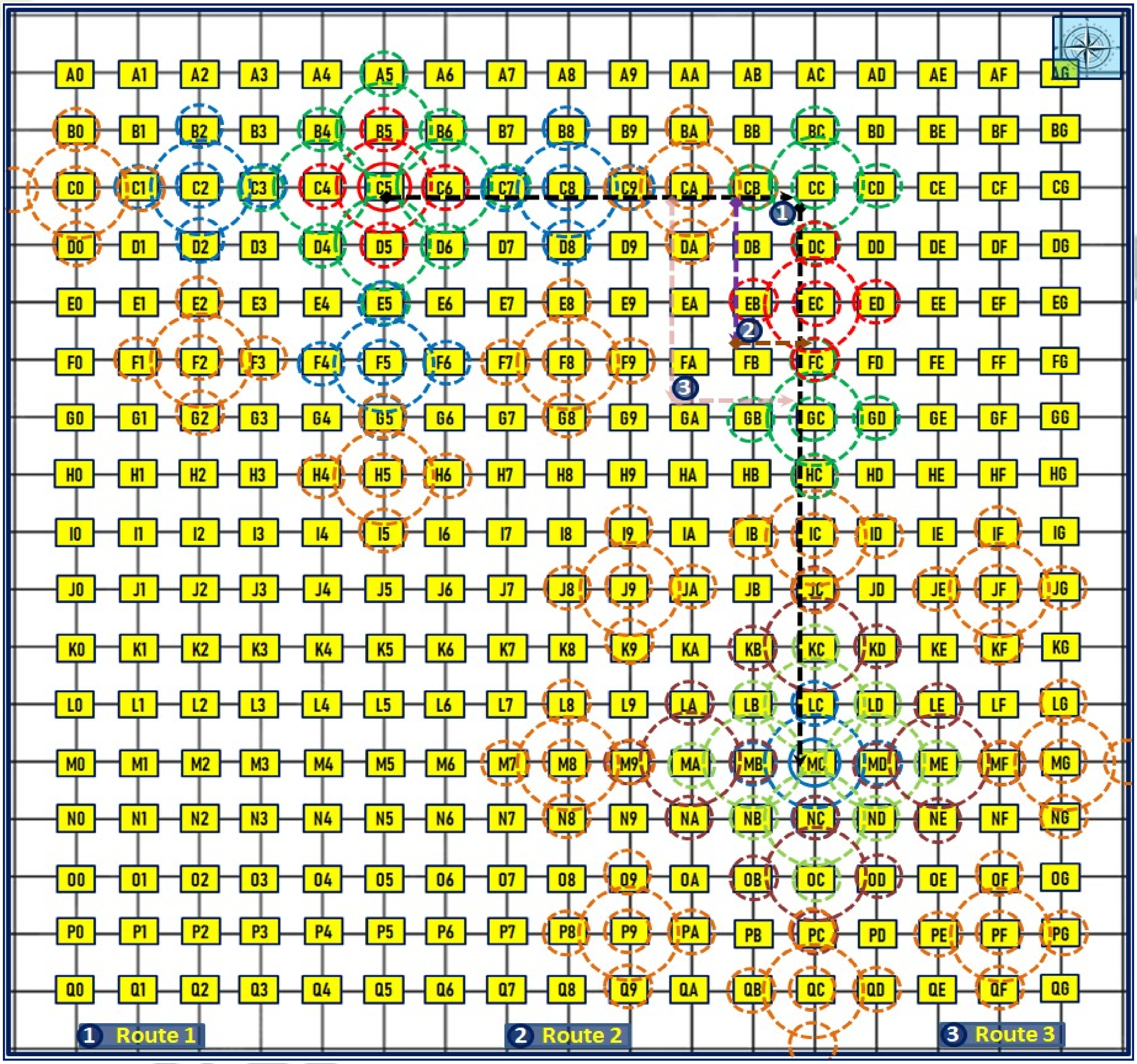


Figure-3: FOUND 3 ROUTES W/ STANDARD G-ROUTING ALGORITHM

ROUTE STEPS	NODE	DIRECTION	DISTANCE	ROUTE STEPS	NODE	DIRECTION	DISTANCE	ROUTE STEPS	NODE	DIRECTION	DISTANCE
1/1	C5	START	0.00	2/1	C5	START	0.00	3/1	C5	START	0.00
1/2	C6	E	dC6	2/2	C6	E	dC6	3/2	C6	E	dC6
1/3	C7	E	dC7	2/3	C7	E	dC7	3/3	C7	E	dC7
1/4	C8	E	dC8	2/4	C8	E	dC8	3/4	C8	E	dC8
1/5	C9	E	dC9	2/5	C9	E	dC9	3/5	C9	E	dC9
1/6	CA	E	dCA	2/6	CA	E	dCA	3/6	CA	E	dCA
1/7	CB	E	dCB	2/7	CB	E	dCB	3/7	DA	S	dDA
1/8	CC	E	dCC	2/8	DB	S	dDB	3/8	EA	S	dEA
1/9	DC	S	dDC	2/9	EB	S	dEB	3/9	FA	S	dFA
1/10	EC	S	dEC	2/10	FB	S	dFB	3/10	GA	S	dGA
1/11	FC	S	dFC	2/11	FC	E	dFC	3/11	GB	E	dGB
1/12	GC	S	dGC	2/12	GC	S	dGC	3/12	GC	E	dGC
1/13	HC	S	dHC	2/13	HC	S	dHC	3/13	HC	S	dHC
1/14	IC	S	dIC	2/14	IC	S	dIC	3/14	IC	S	dIC
1/15	JC	S	dJC	2/15	JC	S	dJC	3/15	JC	S	dJC
1/16	KC	S	dKC	2/16	KC	S	dKC	3/16	KC	S	dKC
1/17	LC	S	dLC	2/17	LC	S	dLC	3/17	LC	S	dLC
1/18	MC	S	dMC	2/18	MC	S	dMC	3/18	MC	S	dMC
Steps := 18	No. Turns := 1 (E,S)		Total Distance	Steps := 18	No. Turns := 3 (E,S,E,S)		Total Distance	Steps := 18	No. Turns := 3 (E,S,E,S)		Total Distance

Table-3: FOUND 3 ROUTES W/ STANDARD G-ROUTING ALGORITHM

Optimization G-ROUTING ALGORITHM Methodology

The optimization G-ROUTING ALGORITHM can be applied to uniform geometry network layouts like squares layout and hexagon layout or any other uniform geometry layouts. **Figure-1** shows the squares uniform layout, the optimization can be quickly search for crossing quadrant from source and destination nodes by starting with clockwise in outward direction with 90° turns at 12 O' clock with only North, East, and South then West directions. The sequences below show step-by-step for the optimization routing.

1. Start with source node '**C5**' on the **Left-Side**, there are 4 neighbor nodes '**B5, C6, D5, C4**' in order of clockwise direction start at 12 O' clock; the following step '1.1' starts with 1st node '**B5**' with the outward directions has neighbor node '**A5**'; the next step '1.2' starts with 2nd node '**C6**' with the outward directions has neighbor node '**C7**'; the next steps would be the same for the other nodes '**D5**' with the neighbor node '**E5**' and '**C4**' with neighbor node '**C3**' in steps '1.3' and '1.4'.
2. Start with destination node '**MC**' on the **Right-Side**, there are 4 neighbor nodes '**LC, MD, NC, MB**' in order of clockwise direction start at 12 O' clock; the following step '1.1' starts with 1st node '**LC**' with the outward direction neighbor node '**KC**'; the

next step '1.2' starts with 2nd node '**MD**' with outward direction neighbor node '**ME**'; the next steps would be the same for the other nodes '**NC**' with the neighbor node '**OC**' and '**MB**' with neighbor node '**MA**' in steps '1.3' and '1.4'.

3. Continue from both sides step-by-step with the sequences shown in **Table-5**, the optimization algorithm found shared node '**M5**' in South direction of the source node '**C5**' and in West direction of destination node '**MC**'; found another shared node '**CC**' in East direction of the source node '**C5**' and in North direction of destination node '**MC**'. These 2 shared nodes will create the 2 routes from the source to the destination node.
4. With this quick methodology, the algorithm found the crossing quadrant with 2 fastest routes around this quadrant. **Table-4** shows these 2 routes with the same number of routing nodes and the same number of turns. This optimization is perfect routing method for the squares uniform network layout like Satellite Network for Internet StarLink Network.
5. **Figure-4** below shows graphical layout and directions of this optimization G-ROUTING ALGORITHM for the squares uniform network layout.

ROUTE STEPS	NODE	DIRECTION	DISTANCE	ROUTE STEPS	NODE	DIRECTION	DISTANCE
1/1	C5	START	0.00	2/1	C5	START	0.00
1/2	D5	S	dD5	2/2	C6	E	dC6
1/3	E5	S	dE5	2/3	C7	E	dC7
1/4	F5	S	dF5	2/4	C8	E	dC8
1/5	G5	S	dG5	2/5	C9	E	dC9
1/6	H5	S	dH5	2/6	CA	E	dCA
1/7	I5	S	dI5	2/7	CB	E	dCB
1/8	J5	S	dJ5	2/8	CC	E	dCC
1/9	K5	S	dK5	2/9	DC	S	dDC
1/10	L5	S	dL5	2/10	EC	S	dEC
1/11	M5	E	dM5	2/11	FC	S	dFC
1/12	M6	E	dM6	2/12	GC	S	dGC
1/13	M7	E	dM7	2/13	HC	S	dHC
1/14	M8	E	dM8	2/14	IC	S	dIC
1/15	M9	E	dM9	2/15	JC	S	dJC
1/16	MA	E	dMA	2/16	KC	S	dKC
1/17	MB	E	dMB	2/17	LC	S	dLC
1/18	MC	E	dMC	2/18	MC	S	dMC
Steps := 18	No. Turns := 1 (S,E)		Total Distance	Steps := 18	No. Turns := 1 (E,S)		Total Distance

Table-4: FOUND 2 ROUTES W/ OPTIMIZATION G-ROUTING ALGORITHM

henryvpham@gmail.com

START SOURCE NODE 'C5'			START DESTINATION NODE 'MC'		
Steps	Node	Neighbor Nodes	Steps	Node	Neighbor Nodes
1	C5	B5, C6, D5, C4	1	MC	LC, MD, NC, MB
1.1	B5	A5	1.1	LC	KC
1.2	C6	C7	1.2	MD	ME
1.3	D5	E5	1.3	NC	OC
1.4	C4	C3	1.4	MB	MA
1.1.1	A5	xxx	1.1.1	KC	JC
1.2.1	C7	C8	1.2.1	ME	MF
1.3.1	E5	F5	1.3.1	OC	PC
1.4.1	C3	C2	1.4.1	MA	M9
1.2.1.1	C8	C9	1.1.1.1	JC	IC
1.3.1.1	F5	G5	1.2.1.1	MF	MG
1.4.1.1	C2	C1	1.3.1.1	PC	QC
1.2.1.1.1	C9	CA	1.4.1.1	M9	M8
1.3.1.1.1	G5	H5	1.1.1.1.1	IC	HC
1.4.1.1.1	C1	C0	1.2.1.1.1	MG	xxx
1.2.1.1.1.1	CA	CB	1.3.1.1.1	QC	xxx
1.3.1.1.1.1	H5	I5	1.4.1.1.1	M8	M7
1.4.1.1.1.1	C0	xxx	1.1.1.1.1.1	HC	GC
1.2.1.1.1.1.1	CB	CC	1.4.1.1.1.1	M7	M6
1.3.1.1.1.1.1	I5	J5	1.1.1.1.1.1.1	GC	FC
1.2.1.1.1.1.1.1	CC	CD	1.4.1.1.1.1.1	M6	M5
1.3.1.1.1.1.1.1	J5	K5	1.1.1.1.1.1.1.1	FC	EC
1.2.1.1.1.1.1.1.1	CD	CE	1.4.1.1.1.1.1.1	M5	M4
1.3.1.1.1.1.1.1.1	K5	L5	1.1.1.1.1.1.1.1.1	EC	DC
1.2.1.1.1.1.1.1.1.1	CE	CF	1.4.1.1.1.1.1.1.1	M4	M3
1.3.1.1.1.1.1.1.1.1	L5	M5	1.1.1.1.1.1.1.1.1.1	DC	CC
SQUARE SEARCHING CROSSING QUADRANT G-ROUTE IN 90° OUTWARD DIRECTIONS					

Table-5: FAST SEARCH 90° OUTWARD DIRECTIONS G-ROUTING ALGORITHM

G-ROUTING ALGORITHM METHODOLOGY

Henry V. Pham

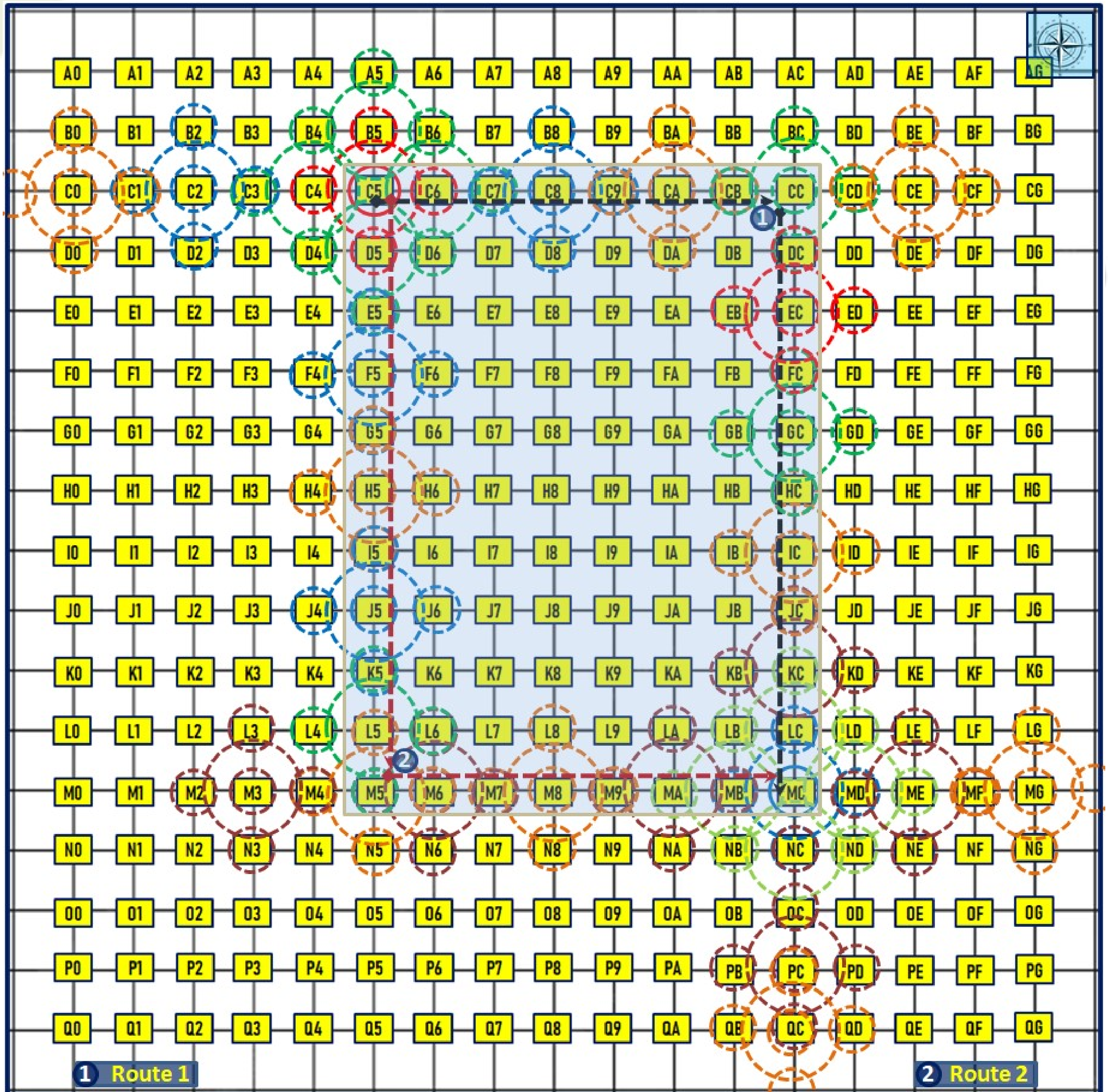


Figure-4: FOUND 2 ROUTES W/ OPTIMIZATION G-ROUTING ALGORITHM

Introduce Diagonals in Square Layout

The squares uniform network layout is perfect for optimization G-ROUTING ALGORITHM. However, with future squares uniform network layouts or the Traffic Lights Routing for "**Emergency Traffic Lights Routing System**", the algorithm needs to combine both optimization and standard routing methods to search for fastest routes within the crossed quadrant network section. This section will introduce the two diagonal lines and an inner square with four lines in **Figure-2**, and the G-ROUTING algorithm will find the fastest route goes through the diagonal line as a shortcut with combination of searching for crossed quadrant and searching within the crossed quadrant methods. The optimization method above already found the crossed quadrant section of the square network layout in **Figure-1**. Now, let's use rule number 7 to search for the fastest route in **Figure-2** within the crossed quadrant by applying the standard G-ROUTING ALGORITHM method in clockwise direction for both source and destination nodes within the crossed section toward each other direction in degree angles within the crossed section.

1. Start with source node '**C5**' on the **Left-Side** and based on the database **Table-1**, there are 2 neighbor nodes '**C6, D5**' in order of clockwise direction start at 12 O' clock within the crossed quadrant and within **East and South** directions; the following step '1.1' starts with 1st node '**C6**' with 90° SE (South-East) neighbor nodes '**C7**, **D6**'; the next step '1.2' starts with 2nd node '**D5**' with 90° SE (South-East) neighbor nodes '**D6, E5**' with duplicated node '**D6**' from the previous steps; the next steps would be the same for the other nodes '**C7**' with the neighbor nodes '**C8, D7**' and node '**D6**' with the neighbor nodes '**D7, E6**' with duplicated node '**D7**' from the previous steps in steps '1.1.1' and '1.1.2' for nodes '**C7**' and '**D6**'.
2. Start with destination node '**MC**' on the **Right-Side**, there are 3 neighbor nodes '**LC, MB, LB**' in order of clockwise direction start at 12 O' clock within the crossed quadrant and within **North and West** directions; the following step '1.1' starts with 1st node '**LC**' with 90° NW (North-West) has neighbor nodes '**KC, LB**'; the next step '1.2' starts with 2nd node '**MB**' with 90° NW (North-West) has neighbor nodes '**LB, MA**' with duplicated node '**LB**'; the next steps would be the same for the other

Pham, Henry V.
henryvpham@gmail.com

nodes '**LB**' with the neighbor nodes '**KB, LA, KA**' and '**KC**' with the neighbor node '**JC, KB**' with duplicated node '**KB**' in steps '1.3' and '1.1.1'.

3. Continue from both sides step-by-step with the sequences shown in **Table-6**, the algorithm found the first shared node '**I8**' from direction of the source node '**C5**' at step '1.2.2.2.2.2' and at step '1.3.3.3' in direction from the destination node '**MC**'; the following iterations found another shared node '**H8**' at step '1.2.2.2.2.2' in direction from the source node '**C5**' and at step '1.3.3.3.3' in direction from the destination node '**MC**'; and found another shared node '**H7**' at step '1.2.2.2.2' in direction from the source node '**C5**' and at step '1.3.3.3.3' in direction from the destination node '**MC**'. These 3 shared nodes will create the 3 routes from the source to the destination node; given 3 routes are good enough to compare and to choose the best one.
4. Now, links the 1st shared node '**I8**' to find the 1st route. Starts from the **Left-Side**, node '**I8**' is connected to '**H7**', node '**H7**' is connected to node '**G6**', node '**G6**' is connected to '**F5**', node '**F5**' is connected to '**E5**', node '**E5**' is connected to '**D5**', node '**D5**' is connected to the source node '**C5**'. Then starts from the **Right-Side**, node '**I8**' is connected to '**J9**', node '**J9**' is connected to '**KA**', node '**KA**' is connected to '**LB**', node '**LB**' is connected to the destination node '**MC**'. So, the 1st route would be in the following series of nodes '**C5**, **D5, E5, F5, G6, H7, I8, J9, KA, LB, MC**'; shared node '**H8**' creates the 2nd route with the following series of nodes '**C5**, **D5, E5, F5, G6, H7, H8, I8, J9, KA, LB, MC**'; and shared node '**H7**' creates the 3rd route with the following series of nodes '**C5**, **D5, E5, F5, G6, H7, I8, J9, KA, LB, MC**'. **Table-7** below shows these 3 routes for comparison. The 1st route and the 3rd route are the best routes to choose with the same routing nodes and with only 1 turn; the programming algorithm can check further for more routes if requires to have more routing options to choose and eliminate the duplicated routes.
5. Finally, **Figure-5** below shows the graphical directions for all 3 routes above. The G-ROUTING ALGORITHM produces the best route in **Figure-5** to compare with the best route in **Figure-3** above with introduced of diagonal lines; this route in **Figure-5** falls into the diagonal line as a shortcut with only 11 routing nodes compare to 18 routing nodes in **Figure-3** without diagonal lines.

START SOURCE NODE 'C5'			START DESTINATION NODE 'MC'		
Steps	Node	Neighbor Nodes	Steps	Node	Neighbor Nodes
1	C5	C6, D5	1	MC	LC, MB, LB
1.1	C6	C7, D6	1.1	LC	KC, LB
1.2	D5	D6, E5	1.2	MB	LB, MA
1.1.1	C7	C8, D7	1.3	LB	KB, LA, KA
1.1.2	D6	D7, E6	1.1.1	KC	JC, KB
1.2.2	E5	E6, F5	1.2.2	MA	LA, M9
1.1.1.1	C8	C9, D8	1.3.1	KB	JB, KA
1.1.1.2	D7	D8, E7	1.3.2	LA	KA, L9
1.1.2.2	E6	E7, F6	1.3.3	KA	JA, K9, J9
1.2.2.2	F5	F6, G6, G5	1.1.1.1	JC	IC, JB
1.1.1.1.1	C9	CA, D9	1.2.2.2	M9	L9, M8
1.1.1.1.2	D8	D9, E8	1.2.2.2	JB	IB, JA
1.1.1.2.2	E7	E8, F7	1.3.2.2	L9	K9, L8
1.1.2.2.2	F6	F7, G6	1.3.3.1	JA	IA, J9
1.2.2.2.2	G6	G7, H7, H6	1.3.3.2	K9	J9, K8
1.2.2.2.3	G5	G6, H5	1.3.3.3	J9	I9, J8, I8
1.1.1.1.1.1	CA	CB, DA	1.1.1.1.1	IC	HC, IB
1.1.1.1.1.2	D9	DA, E9	1.2.2.2.2	M8	L8, M7
1.1.1.1.2.2	E8	E9, F8	1.2.2.2.1	IB	HB, IA
1.1.1.2.2.2	F7	F8, G7	1.3.2.2.2	L8	K8, L7
1.2.2.2.2.1	G7	G8, H7	1.3.3.1.1	IA	HA, I9
1.2.2.2.2.2	H7	H8, I8, I7	1.3.3.2.2	K8	J8, K7
1.2.2.2.2.3	H6	H7, I6	1.3.3.3.1	I9	H9, I8
1.2.2.2.3.2	H5	H6, I5	1.3.3.3.2	J8	I8, J7
1.1.1.1.1.1.1	CB	CC, DB	1.3.3.3.3	I8	H8, I7, H7
SQUARE SEARCHING CROSSED QUADRANT G-ROUTE SEQUENCE					

Table-6: SEARCH WITHIN CROSSED QUADRANT W/ DIAGONALS ROUTING ALGORITHM

ROUTE STEPS	NODE	DIRECTION	DISTANCE	ROUTE STEPS	NODE	DIRECTION	DISTANCE	ROUTE STEPS	NODE	DIRECTION	DISTANCE
1/1	C5	START	0.00	2/1	C5	START	0.00	3/1	C5	START	0.00
1/2	D5	S	dD5	2/2	D5	S	dD5	3/2	D5	S	dD5
1/3	E5	S	dE5	2/3	E5	S	dE5	3/3	E5	S	dE5
1/4	F5	S	dF5	2/4	F5	S	dF5	3/4	F5	S	dF5
1/5	G6	SE	dG6	2/5	G6	SE	dG6	3/5	G6	SE	dG6
1/6	H7	SE	dH7	2/6	H7	SE	dH7	3/6	H7	SE	dH7
1/7	I8	SE	dI8	2/7	H8	E	dH8	3/7	I8	SE	dI8
1/8	J9	SE	dJ9	2/8	I8	S	dI8	3/8	J9	SE	dJ9
1/9	KA	SE	dKA	2/9	J9	SE	dJ9	3/9	KA	SE	dKA
1/10	LB	SE	dLB	2/10	KA	SE	dKA	3/10	LB	SE	dLB
1/11	MC	SE	dMC	2/11	LB	SE	dLB	3/11	MC	SE	dMC
				2/12	MC	SE	dMC				
Steps := 11	No. Turns := 1 (S,SE)		Total Distance	Steps := 12	No. Turns:=4 (S,SE,E,S,SE)		Total Distance	Steps := 11	No. Turns:=1 (S,SE)		Total Distance

Table-7: FOUND 3 ROUTES W/ OPTIMIZATION W/ DIAGONALS G-ROUTING ALGORITHM

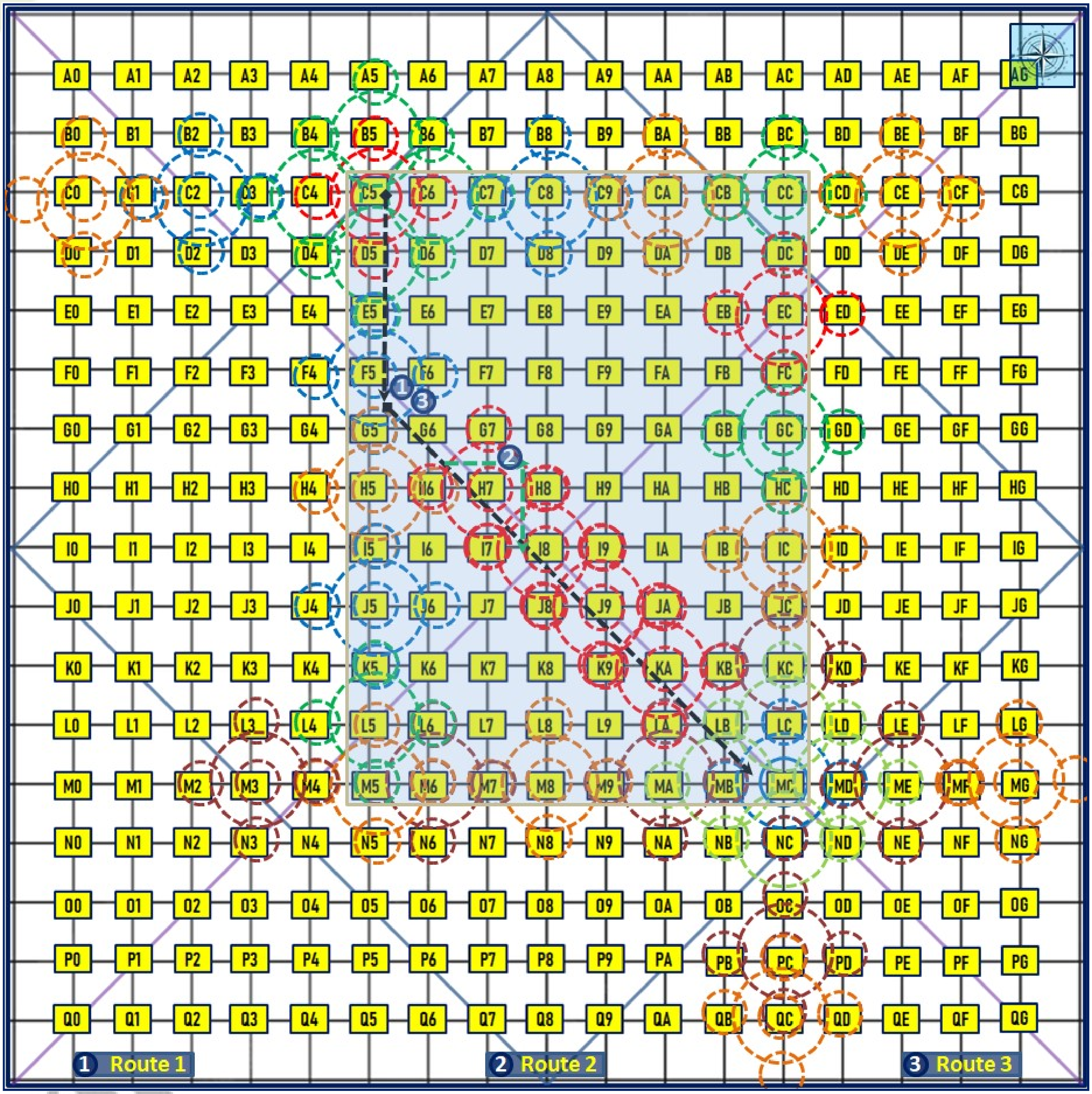


Figure-5: FOUND 3 ROUTES W/ OPTIMIZATION W/ DIAGONALS G-ROUTING ALGORITHM

henryvpham@gmail.com

Introduce Disconnected Nodes Network Layout

In reality, the networks could have some nodes are disconnected from the network for any reasons, and the G-ROUTING ALGORITHM has the capability to create routes with isolating disconnected nodes. Let's start with disconnected nodes '**H8, H9, HA, HB, HC, HD**' and '**I8, I9, IA, IB, IC, ID**' as shown in **Figure-6**, and use optimization rule 6 and rule 3 to search for crossing section. The sequences below show step-by-step to search for the routes in disconnected nodes network layout with optimization routing.

1. Start with source node '**C5**' on the **Left-Side**, there are 4 neighbor nodes '**B5, C6, D5, C4**' in order of clockwise direction start at 12 O' clock; the following step '1.1' starts with 1st node '**B5**' with the outward direction neighbor node '**A5**'; the next step '1.2' starts with 2nd node '**C6**' with the outward direction neighbor node '**C7**'; the next steps would be the same for the other nodes '**D5**' with the neighbor node '**E5**' and '**C4**' with neighbor node '**C3**' in steps '1.3' and '1.4'.
2. Start with destination node '**MC'** on the **Right-Side**, there are 4 neighbor nodes '**LC, MD, NC, MB**' in order of clockwise direction start at 12 O' clock; the following step '1.1' starts with 1st node '**LC**' with the outward direction neighbor node '**KC**'; the next step '1.2' starts with 2nd node '**MD**' with outward direction neighbor node '**ME**'; the next steps would be the same for the other nodes '**NC**' with the neighbor node '**OC**' and '**MB**' with neighbor node '**MA**' in steps '1.3' and '1.4'.
3. Continue from both sides step-by-step with the sequences shown in **Table-8**, when the path from destination node '**MC**' hits node '**JC**' which has disconnected North neighbor node '**IC**'; node '**JC**' will look for clockwise available neighbor node '**JD**'; then next iterations node '**JD**' still has disconnected North neighbor node '**ID**' and looks for clockwise available neighbor node '**JE**'; and finally next iterations node '**JE**' has its North available neighbor node '**IE**' to get back the original outward North direction and later node '**IE**' will keep its North outward direction until this direction crosses East direction node of the source node '**C5**'.
4. Continue from both sides step-by-step with the sequences shown in **Table-8**, the optimization algorithm found shared node '**M5**' in South direction of the source node '**C5**' and in West direction of destination node '**MC**'; found another shared node '**CE**' in East direction of the source node '**C5**' and in North direction of

henryvpham@gmail.com

destination node '**MC**'. These 2 shared nodes will create the 2 routes from the source to the destination node.

5. With this quick methodology, the algorithm found the crossing quadrant with 2 routes around this quadrant. **Table-9** shows these 2 routes; 1st route for shared node '**M5**' with series of 18 routing nodes '**C5**, **D5, E5, F5, G5, H5, I5, J5, K5, L5, M5, M6, M7, M8, M9, MA, MB, MC**'; and 2nd route for shared node '**CE**' with series of 22 routing nodes '**C5**, **C6, C7, C8, C9, CA, CB, CC, CD, CE, DE, EE, FE, GE, HE, IE, JE, JD, JC, KC, LC, MC**'. This optimization is perfect routing method for the squares uniform network layout like Satellite Network for Internet StarLink Network.
6. **Figure-6** below shows graphical layout and directions of this optimization G-ROUTING ALGORITHM for optimization routing of searching for crossed section quadrant of the squares uniform network layout.

START SOURCE NODE 'C5'				START DESTINATION NODE 'MC'		
Steps	Node	Neighbor Nodes		Steps	Node	Neighbor Nodes
1	C5	B5, C6, D5, C4		1	MC	LC, MD, NC, MB
1.1	B5	A5		1.1	LC	KC
1.2	C6	C7		1.2	MD	ME
1.3	D5	E5		1.3	NC	OC
1.4	C4	C3		1.4	MB	MA
1.1.1	A5	xxx		1.1.1	KC	JC
1.2.1	C7	C8		1.2.1	ME	MF
1.3.1	E5	F5		1.3.1	OC	PC
1.4.1	C3	C2		1.4.1	MA	M9
1.2.1.1	C8	C9		1.1.1.1	JC	JD
1.3.1.1	F5	G5		1.2.1.1	MF	MG
1.4.1.1	C2	C0		1.3.1.1	PC	QC
1.2.1.1.1	C9	CA		1.4.1.1	M9	M8
1.3.1.1.1	G5	H5		1.1.1.1.1	JD	JE
1.4.1.1.1	C0	xxx		1.2.1.1.1	MG	xxx
1.2.1.1.1.1	CA	CB		1.3.1.1.1	QC	xxx
1.3.1.1.1.1	H5	I5		1.4.1.1.1	M8	M7
1.2.1.1.1.1.1	CB	CC		1.1.1.1.1.1	JE	IE
1.3.1.1.1.1.1	I5	J5		1.4.1.1.1.1	M7	M6
1.2.1.1.1.1.1.1	CC	CD		1.1.1.1.1.1.1	IE	HE
1.3.1.1.1.1.1.1	J5	K5		1.4.1.1.1.1.1	M6	M5
1.2.1.1.1.1.1.1.1	CD	CE		1.1.1.1.1.1.1.1	HE	GE
1.3.1.1.1.1.1.1.1	K5	L5		1.4.1.1.1.1.1.1	M5	<CROSSED>
1.2.1.1.1.1.1.1.1.1	CE	CF		1.1.1.1.1.1.1.1.1	GE	FE
1.3.1.1.1.1.1.1.1.1	L5	M5		1.1.1.1.1.1.1.1.1.1	FE	EE
1.2.1.1.1.1.1.1.1.1.1	CF	CG		1.1.1.1.1.1.1.1.1.1.1.	EE	DE
1.3.1.1.1.1.1.1.1.1.1	M5	<CROSSED>		1.1.1.1.1.1.1.1.1.1.1.	DE	CE
DISCONNECTED NODES SEARCHING CROSSING QUADRANT G-ROUTE IN 90° OUTWARD DIRECTIONS						

Table-8: FAST SEARCH 90° OUTWARD DIRECTIONS W/ DISCONNECTED NODES G-ROUTING ALGORITHM

ROUTE STEPS	NODE	DIRECTION	DISTANCE	ROUTE STEPS	NODE	DIRECTION	DISTANCE
1/1	C5	START	0.00	2/1	C5	START	0.00
1/2	D5	S	dD5	2/2	C6	E	dC6
1/3	E5	S	dE5	2/3	C7	E	dC7
1/4	F5	S	dF5	2/4	C8	E	dC8
1/5	G5	S	dG5	2/5	C9	E	dC9
1/6	H5	S	dH5	2/6	CA	E	dCA
1/7	I5	S	dI5	2/7	CB	E	dCB
1/8	J5	S	dJ5	2/8	CC	E	dCC
1/9	K5	S	dK5	2/9	CD	E	dCD
1/10	L5	S	dL5	2/10	CE	E	dCE
1/11	M5	E	dM5	2/11	DE	S	dDE
1/12	M6	E	dM6	2/12	EE	S	dEE
1/13	M7	E	dM7	2/13	FE	S	dFE
1/14	M8	E	dM8	2/14	GE	S	dGE
1/15	M9	E	dM9	2/15	HE	S	dHE
1/16	MA	E	dMA	2/16	IE	S	dIE
1/17	MB	E	dMB	2/17	JE	S	dJE
1/18	MC	E	dMC	2/18	JD	W	dJD
				2/19	JC	W	dJC
				2/20	KC	S	dKC
				2/21	LC	S	dLC
				2/22	MC	S	dMC
Steps := 18	No. Turns := 1 (S,E)		Total Distance	Steps := 22	No. Turns := 3 (E,S,W,S)		Total Distance

Table-9: FOUND 2 ROUTES W/ OUTWARD DIRECTIONS W/ DIAGONALS W/ DISCONNECTED NODES G-ROUTING ALGORITHM

To complete this routing within the crossed quadrant section with disconnected nodes, the following sequences show step-by-step to search for the fastest route within the crossed section with disconnected nodes ‘**H8, H9, HA, HB, HC, HD**’ and ‘**I8, I9, IA, IB, IC, ID**’ as shown in **Table-10**.

1. Start with source node ‘**C5**’ on the **Left-Side**, there are 2 neighbor nodes ‘**C6, D5**’ in order of clockwise direction start at 12 O’ clock within the crossed quadrant and within **East and South** directions; the following step ‘1.1’ starts with 1st node ‘**C6**’ with 90° SE (South-East) has neighbor nodes ‘**C7**, **D6**’; the next step ‘1.2’ starts with 2nd node ‘**D5**’ with 90° SE (South-East) has neighbor nodes ‘**D6, E5**’ with duplicated node ‘**D6**’ from the previous steps; the next steps would be the same for the other nodes ‘**C7**’ with the neighbor nodes ‘**C8, D7**’ and node ‘**D6**’ with the neighbor nodes

'**D7, E6**' with duplicated node '**D7**' from the previous steps in steps '1.1.1' and '1.1.2' for nodes '**C7**' and '**D6**'.

2. Start with destination node '**MC**' on the **Right-Side**, there are 3 neighbor nodes '**LC, MB, LB**' in order of clockwise direction start at 12 O' clock within the crossed quadrant and within **North and West** directions; the following step '1.1' starts with 1st node '**LC**' with 90° NW (North-West) has neighbor nodes '**KC, LB**'; the next step '1.2' starts with 2nd node '**MB**' with 90° NW (North-West) has neighbor nodes '**LB, MA**' with duplicated node '**LB**'; the next steps would be the same for the other nodes '**LB**' with the neighbor nodes '**KB, LA, KA**' and '**KC**' with the neighbor node '**JC, KB**' with duplicated node '**KB**' in steps '1.3' and '1.1.1'.
3. Continue from both sides step-by-step with the sequences shown in **Table-10,** similar to the sequences shown in **Table-6**, except eliminating disconnected nodes when looking for the neighbor nodes within the crossed section and with the toward angle direction 90° SE (**South-East)** to the destination node '**MC**' or 90° NW (**North-West**) to the source node '**C5**'. As shown in **Table-10**, the algorithm found the 1st shared node '**I7**' at step '1.2.2.2.2.2' in direction from the source node '**C5**' and at step '1.3.3.3.1.1' in direction from the destination node '**MC**'; found the 2st shared node '**H7**' at step '1.2.2.2.2' in direction from the source node '**C5**' and at step '1 1.3.3.3.1.1.1' in direction from the destination node '**MC**'; and found the 3st shared node '**I6**' at step '1.2.2.2.2.3' in direction from the source node '**C5**' and at step '1 1.3.3.3.1.1.1' in direction from the destination node '**MC**'. These 3 shared nodes will create the 3 routes from the source to the destination node; given 3 routes are good enough to compare and to choose the best one.
4. Finally, links these shared nodes with their paths together we have these 3 routes which are shown in **Table-11** below with the best route with series of routing nodes '**C5**, **D5, E5, F5, G6, H7, I7, J7, J8, J9, KA, LB, MC**' with 4 turns (S, SE, S, E, SE) directions. **Figure-6** shows graphical directions for all 3 routes above within the crossed quadrant section with disconnected nodes '**H8, H9, HA, HB, HC, HD**' and '**I8, I9, IA, IB, IC, ID**'.

START SOURCE NODE 'C5'				START DESTINATION NODE 'MC'		
Steps	Node	Neighbor Nodes		Steps	Node	Neighbor Nodes
1	C5	C6, D5		1	MC	LC, MB, LB
1.1	C6	C7, D6		1.1	LC	KC, LB
1.2	D5	D6, E5		1.2	MB	LB, MA
1.1.1	C7	C8, D7		1.3	LB	KB, LA, KA
1.1.2	D6	D7, E6		1.1.1	KC	JC, KB
1.2.2	E5	E6, F5		1.2.2	MA	LA, M9
1.1.1.1	C8	C9, D8		1.3.1	KB	JB, KA
1.1.1.2	D7	D8, E7		1.3.2	LA	KA, L9
1.1.2.2	E6	E7, F6		1.3.3	KA	JA, K9, J9
1.2.2.2	F5	F6, G6, G5		1.1.1.1	JC	JD, JB (JD:Boundary)
1.1.1.1.1	C9	CA, D9		1.2.2.2	M9	L9, M8
1.1.1.1.2	D8	D9, E8		1.3.1.1	JB	JA
1.1.1.2.2	E7	E8, F7		1.3.2.2	L9	K9, L8
1.1.2.2.2	F6	F7, G6		1.3.3.1	JA	J9
1.2.2.2.2	G6	G7, H7, H6		1.3.3.2	K9	J9, K8
1.2.2.2.3	G5	G6, H5		1.3.3.3	J9	J8
1.1.1.1.1.1	CA	CB, DA		1.1.1.1.1	JD	JE, JC (JE: Boundary)
1.1.1.1.1.2	D9	DA, E9		1.2.2.2.2	M8	L8, M7
1.1.1.1.2.2	E8	E9, F8		1.3.2.2.2	L8	K8, L7
1.1.1.2.2.2	F7	F8, G7		1.3.3.2.2	K8	J8, K7
1.2.2.2.2.1	G7	G8, H7		1.3.3.3.1	J8	J7
1.2.2.2.2.2	H7	I7		1.1.1.1.1.1	JE	IE, JD
1.2.2.2.2.3	H6	H7, I6		1.2.2.2.2.2	M7	L7, M6
1.2.2.2.3.2	H5	H6, I5		1.3.2.2.2.2	L7	K7, L6
1.1.1.1.1.1.1	CB	CC, DC, DB		1.3.3.2.2.2	K7	J7, K6
1.1.1.1.1.1.2	DA	DB, EA		1.3.3.3.1.1	J7	I7, J6
1.1.1.1.1.2.2	E9	EA, F9		1.1.1.1.1.1.1	IE	HE
1.1.1.1.2.2.2	F8	F9, G8		1.2.2.2.2.2.2	M6	L6, M5
1.2.2.2.2.1.1	G8	G9		1.3.2.2.2.2.2	L6	K6, L5
1.2.2.2.2.2.1	I7	J7		1.3.3.2.2.2.2	K6	J6, K5
1.2.2.2.2.3.2	I6	I7, J6		1.3.3.3.1.1.1	I7	H7, I6
DISCONNECTED NODES SQUARE SEARCHING CROSSED QUADRANT G-ROUTING SEQUENCE						

Table-10: SEARCH WITHIN CROSSED QUADRANT W/ DISCONNECTED NODES G-ROUTING ALGORITHM

G-ROUTING ALGORITHM METHODOLOGY

Henry V. Pham

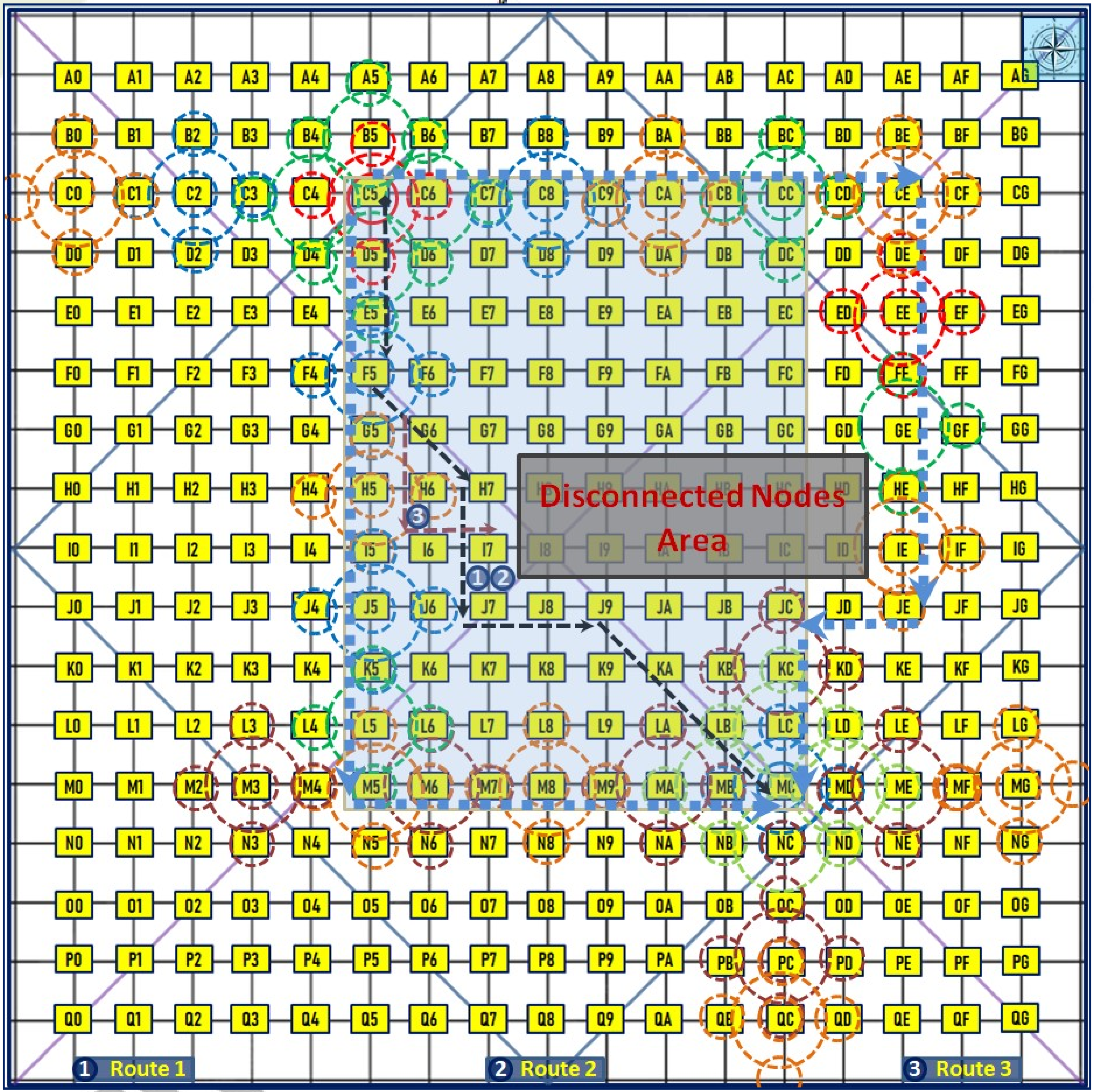


Figure-6: FOUND 3 ROUTES W/ DIAGONALS W/ DISCONNECTED NODES G-ROUTING ALGORITHM

ROUTE STEPS	NODE	DIRECTION	DISTANCE
1/1	C5	START	0.00
1/2	D5	S	dD5
1/3	E5	S	dE5
1/4	F5	S	dF5
1/5	G6	SE	dG6
1/6	H7	SE	dH7
1/7	I7	S	dI7
1/8	J7	S	dJ7
1/9	J8	E	dI8
1/10	J9	E	dJ9
1/11	KA	SE	dKA
1/12	LB	SE	dLB
1/13	MC	SE	dMC
Steps := 13	No. Turns:=4 (S,SE,S,E,SE)		Total Distance

ROUTE STEPS	NODE	DIRECTION	DISTANCE
2/1	C5	START	0.00
2/2	D5	S	dD5
2/3	E5	S	dE5
2/4	F5	S	dF5
2/5	G6	SE	dG6
2/6	H7	SE	dH7
2/7	I7	S	dI7
2/8	J7	S	dJ7
2/9	J8	E	dI8
2/10	J9	E	dJ9
2/11	KA	SE	dKA
2/12	LB	SE	dLB
2/13	MC	SE	dMC
Steps := 13	No. Turns:=4 (S,SE,S,E,SE)		Total Distance

ROUTE STEPS	NODE	DIRECTION	DISTANCE
3/1	C5	START	0.00
3/2	D5	S	dD5
3/3	E5	S	dE5
3/4	F5	S	dF5
3/5	G6	SE	dG6
3/6	H6	S	dH6
3/7	I6	S	dI6
3/8	I7	E	dI7
3/9	J7	S	dJ7
3/10	J8	E	dJ8
3/11	J9	E	dJ9
3/12	KA	SE	dKA
3/13	LB	SE	dLB
3/14	MC	SE	dMC
Steps := 14	No. Turns:=6 (S,SE,S,E,S,E,SE)		Total Distance

Table-11: FOUND 3 ROUTES W/ DIAGONALS W/ DISCONNECTED NODES G-ROUTING ALGORITHM

Introduce G-ROUTING ALGORITHM in Hexagon Layout

Hexagon layout is also an uniform geometry network layout, and the G-ROUTING ALGORITHM works well with this layout as long as the network nodes have their neighbor nodes' properties. **Figure-7** shows an hexagon network layout with each node has maximum of 3 neighbor nodes, and **Table-12** shows each node's neighbor nodes' properties. Each line shows a node with its list of neighbors and its neighbors' properties with distance and direction from that node; the direction of node-to-nodes in **30° bouncing back and forth** within the hexagon layout and this consider one straight path. The table shows node '**B5**' has 3 neighbor nodes '**A2; B6; B4**' in order of clockwise direction start from 12 O' clock. Node '**A2**' has a distance of '**dA2**' in North direction from node '**B5**'; node '**B6**' has a distance of '**dB6**' in **South-East** direction from node '**B5**'; and node '**B4**' has a distance of '**dB4**' in **South-West** direction from node '**B5**'. This node is a sample node from the table database below; other nodes are having the same format of properties.

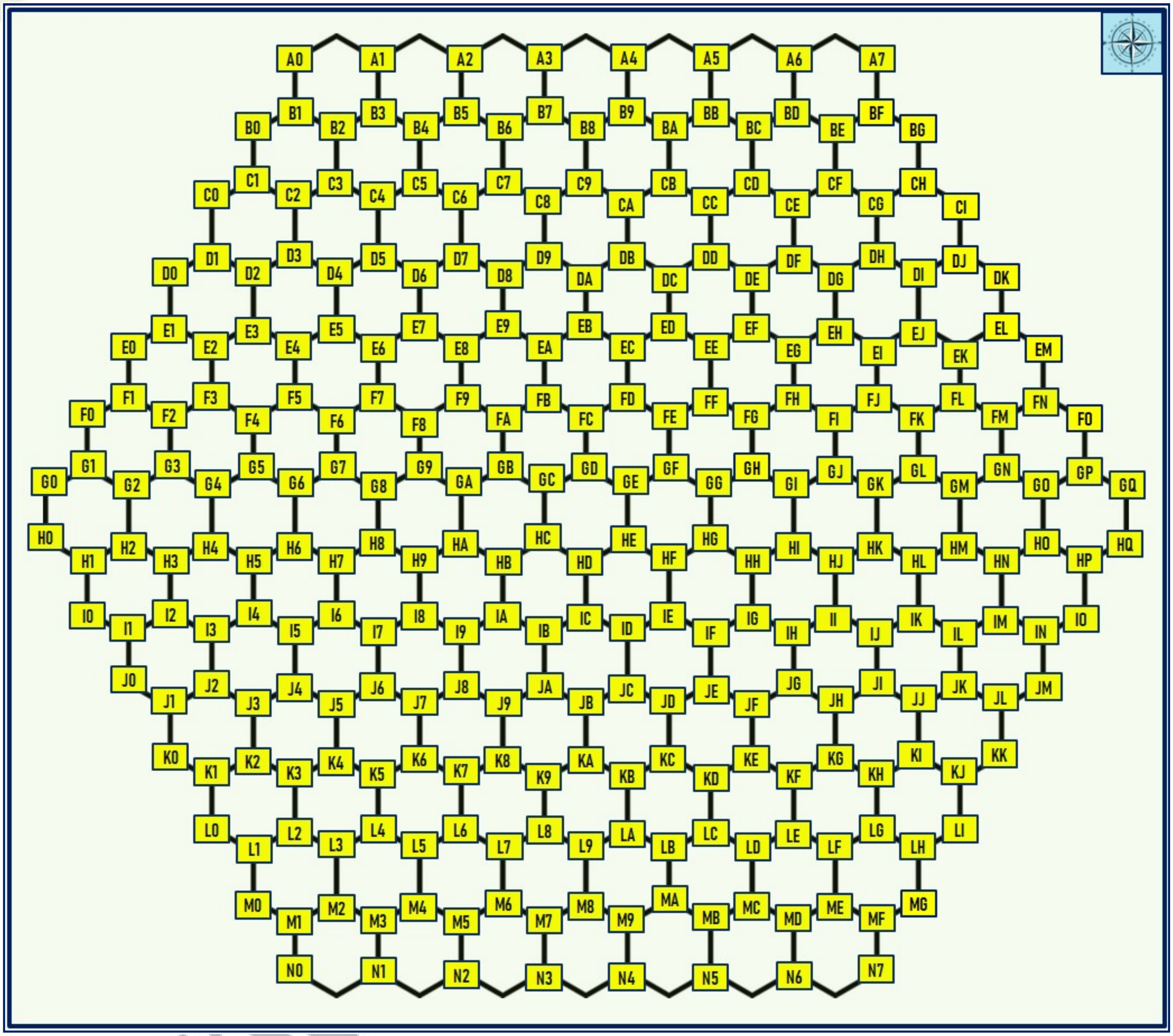


Figure-7: NETWORK NODES IN HEXAGON LAYOUT

NODE	NEIGHBOR NODES	DISTANCE/DIRECTION TO NEIGHBOR NODES (meters)
A0	B1	B1:=[dB1/S]
A1	B3	B3:=[dB3/S]
A2	B5	B5:=[dB5/S]
A3	B7	B7:=[dB7/S]
A4	B9	B9:=[dB9/S]
A5	BB	BB:=[dBB/S]
A6	BD	BD:=[dBD/S]
A7	BF	BF:=[dBF/S]
B0	B1;C1	B1:=[dB1/NE]; C1:=[dC1/S]
B1	A0;B2;B0	A0:=[dA0/N]; B2:=[dB2/SE]; B0:=[dB0/SW]
B2	B3;C3;B1	B3:=[dB3/NE]; C3:=[dC3/S]; B1:=[dB1/NW]
B3	A1;B4;B2	A1:=[dA1/N]; B4:=[dB4/SE]; B2:=[dB2/SW]
B4	B5;C5;B3	B5:=[dB5/NE]; C5:=[dC5/S]; B3:=[dB3/NW]
B5	A2;B6;B4	A2:=[dA2/N]; B6:=[dB6/SE]; B4:=[dB4/SW]
B6	B7;C7;B5	B7:=[dB7/NE]; C7:=[dC7/S]; B5:=[dB5/NW]
B7	A3;B8;B6	A3:=[dA3/N]; B8:=[dB8/SE]; B6:=[dB6/SW]
B8	B9;C9;B7	B9:=[dB9/NE]; C9:=[dC9/S]; B7:=[dB7/NW]
B9	A4;BA;B8	A4:=[dA4/N]; BA:=[dBA/SE]; B8:=[dB8/SW]
BA	BB;CB;B9	BB:=[dBB/NE]; CB:=[dCB/S]; B9:=[dB9/NW]
BB	A5;BC;BA	A5:=[dA5/N]; BC:=[dBC/SE]; BA:=[dBA/SW]
BC	BD;CD;BB	BD:=[dBD/NE]; CD:=[dCD/S]; BB:=[dBB/NW]
BD	A6;BE;BC	A6:=[dA6/N]; BE:=[dBE/SE]; BC:=[dBC/SW]
BE	BF;CF;BD	BE:=[dBE/NE]; CF:=[dCF/S]; BD:=[dBD/NW]
BF	A7;BG;BE	A7:=[dA7/N]; BG:=[dBG/SE]; BE:=[dBE/SW]
BG	CH;BF	CH:=[dCH/S]; BF:=[dBF/NE]
C0	C1;D1	C1:=[dC1/NE]; D1:=[dD1/S]
C1	B0;C2;C0	B0:=[dB0/N]; C2:=[dC2/SE]; C0:=[dC0/SW]
C2	C3;D3;C1	C3:=[dC3/NE]; D3:=[dD3/S]; C1:=[dC1/NW]
C3	B2;C4;C2	B2:=[dB2/N]; C4:=[dC4/SE]; C2:=[dC2/SW]
C4	C5;D5;C3	C5:=[dC5/NE]; D5:=[dD5/S]; C3:=[dC3/NW]
C5	B4;C6;C4	B4:=[dB4/N]; C6:=[dC6/SE]; C4:=[dC4/SW]
C6	C7;D7;C5	C7:=[dC7/NE]; D7:=[dD7/S]; C5:=[dC5/NW]
C7	B6;C8;C6	B6:=[dB6/N]; C8:=[dC8/SE]; C6:=[dC6/SW]
C8	C9;D9;C7	C9:=[dC9/NE]; D9:=[dD9/S]; C7:=[dC7/NW]
C9	B8;CA;C8	B8:=[dB8/N]; CA:=[dCA/SE]; C8:=[dC8/SW]
CA	CB;DB;C9	CB:=[dCB/NE]; DB:=[dDB/S]; C9:=[dC9/NW]
CB	BA;CC;CA	BA:=[dBA/N]; CC:=[dCC/SE]; CA:=[dCA/SW]
CC	CD;DD;CB	CD:=[dCD/NE]; DD:=[dDD/S]; CB:=[dCB/NW]
CD	BC;CE;CC	BC:=[dBC/N]; CE:=[dCE/SE]; CC:=[dCC/SW]
CE	CF;DF;CD	CF:=[dCF/NE]; DF:=[dDF/S]; CD:=[dCD/NW]
CF	BE;CG;CE	BE:=[dBE/N]; CG:=[dCG/SE]; CE:=[dCE/SW]
CG	CH;DH;CF	CH:=[dCH/NE]; DH:=[dDH/S]; CF:=[dCF/NW]
CH	BG;CI;CG	BG:=[dBG/N]; CI:=[dCI/SE]; CG:=[dCG/SW]
CI	DJ;CH	DJ:=[dJD/S]; CH:=[dCH/NW]

D0	D1;E1	D1:=[dD1/NE]; E1:=[dE1/S]
D1	C0;D2;D0	C0:=[dC0/N]; D2:=[dD2/SE]; D0:=[dD0/SW]
D2	D3;E3;D1	D3:=[dD3/NE]; E3:=[dE3/S]; D1:=[dD1/NW]
D3	C2;D4;D2	C2:=[dC2/N]; D4:=[dD4/SE]; D2:=[dD2/SW]
D4	D5;E5;D3	D5:=[dD5/NE]; E5:=[dE5/S]; D3:=[dD3/NW]
D5	C4;D6;D4	C4:=[dC4/N]; D6:=[dD6/SE]; D4:=[dD4/SW]
D6	D7;E7;D5	D7:=[dD7/NE]; E7:=[dE7/S]; D5:=[dD5/NW]
D7	C6;D8;D6	C6:=[dC6/N]; D8:=[dD8/SE]; D6:=[dD6/SW]
D8	D9;E9;D7	D9:=[dD9/NE]; E9:=[dE9/S]; D7:=[dD7/NW]
D9	C8;DA;D8	D8:=[dD8/N]; DA:=[dDA/SE]; D8:=[dD8/SW]
DA	DB;EB;D9	DB:=[dDB/NE]; EB:=[dEB/S]; E9:=[dE9/NW]
DB	CA;DC;DA	DA:=[dDA/N]; DC:=[dDC/SE]; DA:=[dDA/SW]
DC	DD;ED;DB	DD:=[dDD/NE]; ED:=[dED/S]; DB:=[dDB/NW]
DD	CC;DE;DC	CC:=[dCC/N]; DE:=[dDE/SE]; DC:=[dDC/SW]
DE	DF;EF;DD	DF:=[dDF/NE]; EF:=[dEF/S]; DD:=[dDD/NW]
DF	CE;DG;DE	CE:=[dCE/N]; DG:=[dDG/SE]; DE:=[dDE/SW]
DG	DH;EH;DF	DH:=[dDH/NE]; EH:=[dEH/S]; DF:=[dDF/NW]
DH	CG;DI;DG	CG:=[dCG/N]; DI:=[dDI/SE]; DG:=[dDG/SW]
DI	DJ;EJ;DH	DJ:=[dDJ/NE]; EJ:=[dEJ/S]; DH:=[dDH/NW]
DJ	CI;DK;DI	CI:=[dCI/N]; DK:=[dDK/SE]; DI:=[dDI/SW]
DK	EL;DJ	EL:=[dEL/S]; DJ:=[dDJ/NW]
E0	E1;F1	E1:=[dE1/NE]; F1:=[dF1/S]
E1	D0;E2;E0	D0:=[dD0/N]; E2:=[dE2/SE]; E0:=[dE0/SW]
E2	E3;F3;E1	E3:=[dE3/NE]; F3:=[dF3/S]; E1:=[dE1/NW]
E3	D2;E4;E2	D2:=[dD2/N]; E4:=[dE4/SE]; E2:=[dE2/SW]
E4	E5;F5;E3	E5:=[dE5/NE]; F5:=[dF5/S]; E3:=[dE3/NW]
E5	D4;E6;E4	D4:=[dD4/N]; E6:=[dE6/SE]; E4:=[dE4/SW]
E6	E7;F7;E5	E7:=[dE7/NE]; F7:=[dF7/S]; E5:=[dE5/NW]
E7	D6;E8;E6	D6:=[dD6/N]; E8:=[dE8/SE]; E6:=[dE6/SW]
E8	E9;F9;E7	E9:=[dE9/NE]; F9:=[dF9/S]; E7:=[dE7/NW]
E9	D8;EA;E8	D8:=[dD8/N]; EA:=[dEA/SE]; E8:=[dE8/SW]
EA	EB;FB;E9	EB:=[dEB/NE]; FB:=[dFB/S]; E9:=[dE9/NW]
EB	DA;EC;EA	DA:=[dDA/N]; EC:=[dEC/SE]; EA:=[dEA/SW]
EC	ED;FD;EB	ED:=[dED/NE]; FD:=[dFD/S]; EB:=[dEB/NW]
ED	DC;EE;EC	DC:=[dDC/N]; EE:=[dEE/SE]; EC:=[dEC/SW]
EE	EF;FF;ED	EF:=[dEF/NE]; FF:=[dFF/S]; ED:=[dED/NW]
EF	DE;EG;EE	DE:=[dDE/N]; EG:=[dEG/SE]; EE:=[dEE/SW]
EG	EH;FH;EF	EH:=[dEH/NE]; FH:=[dFH/S]; EF:=[dEF/NW]
EH	DG;EI;EG	DG:=[dDG/N]; EI:=[dEI/SE]; EG:=[dEG/SW]
EI	EJ;FJ;EH	EJ:=[dEJ/NE]; FJ:=[dFJ/S]; EH:=[dEH/NW]
EJ	DI;EK;EI	DI:=[dDI/N]; EK:=[dEK/SE]; EI:=[dEI/SW]
EK	EL;FL;EJ	EL:=[dEL/NE]; FL:=[dFL/S]; EJ:=[dEJ/NW]
EL	DK;EM;EK	DK:=[dDK/N]; EM:=[dEM/SE]; EK:=[dEK/SW]
EM	FN;EL	FN:=[dFN/S]; EL:=[dEL/NW]

F0	F1;G1	F1:=[dF1/NE]; G1:=[dG1/S]
F1	E0;F2;F0	E0:=[dE0/N]; F2:=[dF2/SE]; F0:=[dF0/SW]
F2	F3;G3;F1	F3:=[dF3/NE]; G3:=[dG3/S]; F1:=[dF1/NW]
F3	E2;F4;F2	E2:=[dE2/N]; F4:=[dF4/SE]; F2:=[dF2/SW]
F4	F5;G5;F3	F5:=[dF5/NE]; G5:=[dG5/S]; F3:=[dF3/NW]
F5	E4;F6;F4	E4:=[dE4/N]; F6:=[dF6/SE]; F4:=[dF4/SW]
F6	F7;G7;F5	F7:=[dF7/NE]; G7:=[dG7/S]; F5:=[dF5/NW]
F7	E6;F8;F6	E6:=[dE6/N]; F8:=[dF8/SE]; F6:=[dF6/SW]
F8	F9;G9;F7	F9:=[dF9/NE]; G9:=[dG9/S]; F7:=[dF7/NW]
F9	E8;FA;F8	E8:=[dE8/N]; FA:=[dFA/SE]; F8:=[dF8/SW]
FA	FB;GB;F9	FB:=[dFB/NE]; GB:=[dGB/S]; F9:=[dF9/NW]
FB	EA;FC;FA	EA:=[dEA/N]; FC:=[dFC/SE]; FA:=[dFA/SW]
FC	FD;GD;FB	FD:=[dFD/NE]; GD:=[dGD/S]; FB:=[dFB/NW]
FD	EC;FE;FC	EC:=[dEC/N]; FE:=[dFE/SE]; FC:=[dFC/SW]
FE	FF;GF;FD	FF:=[dFF/NE]; GF:=[dGF/S]; FD:=[dFD/NW]
FF	EE;FG;FE	EE:=[dEE/N]; FG:=[dFG/SE]; FE:=[dFE/SW]
FG	FH;GH;FF	FH:=[dFH/NE]; GH:=[dGH/S]; FF:=[dFF/NW]
FH	EG;FI;FG	EG:=[dEG/N]; FI:=[dFI/SE]; FG:=[dFG/SW]
FI	FJ;GJ;FH	FJ:=[dFJ/NE]; GJ:=[dGJ/S]; FH:=[dFH/NW]
FJ	EI;FK;FI	EI:=[dEI/N]; FK:=[dFK/SE]; FI:=[dFI/SW]
FK	FL;GL;FJ	FL:=[dFL/NE]; GL:=[dGL/S]; FJ:=[dFJ/NW]
FL	EK;FM;FK	EK:=[dEK/N]; FM:=[dFM/SE]; FK:=[dFK/SW]
FM	FN;GN;FL	FN:=[dFN/NE]; GN:=[dGN/S]; FL:=[dFL/NW]
FN	EM;FO;FM	EM:=[dEM/N]; FO:=[dFO/SE]; FM:=[dFM/SW]
FO	GP;FN	GP:=[dGP/S]; FN:=[dFN/NW]
G0	G1;H0	G1:=[dG1/NE]; H0:=[dH0/S]
G1	F0;G2;G0	F0:=[dF0/N]; G2:=[dG2/SE]; G0:=[dG0/SW]
G2	G3;H2;G1	G3:=[dG3/NE]; H2:=[dH2/S]; G1:=[dG1/NW]
G3	F2;G4;G2	F2:=[dF2/N]; G4:=[dG4/SE]; G2:=[dG2/SW]
G4	G5;H4;G3	G5:=[dG5/NE]; H4:=[dH4/S]; G3:=[dG3/NW]
G5	F4;G6;G4	F4:=[dF4/N]; G6:=[dG6/SE]; G4:=[dG4/SW]
G6	G7;H6;G5	G7:=[dG7/NE]; H6:=[dH6/S]; G5:=[dG5/NW]
G7	F6;G8;G6	F6:=[dF6/N]; G8:=[dG8/SE]; G6:=[dG6/SW]
G8	G9;H8;G7	G9:=[dG9/NE]; H8:=[dH8/S]; G7:=[dG7/NW]
G9	F8;GA;G8	F8:=[dF8/N]; GA:=[dGA/SE]; G8:=[dG8/SW]
GA	GB;HA;G9	GB:=[dGB/NE]; HA:=[dHA/S]; G9:=[dG9/NW]
GB	FA;GC;GA	FA:=[dFA/N]; GC:=[dGC/SE]; GA:=[dGA/SW]
GC	GD;HC;GB	GD:=[dGD/NE]; HC:=[dHC/S]; GB:=[dGB/NW]
GD	FC;GE;GC	FC:=[dFC/N]; GE:=[dGE/SE]; GC:=[dGC/SW]
GE	GF;HE;GD	GF:=[dGF/NE]; HE:=[dHE/S]; GD:=[dGD/NW]
GF	FE;GG;GE	FE:=[dFE/N]; GG:=[dGG/SE]; GE:=[dGE/SW]
GG	GH;HG;GF	GH:=[dGH/NE]; HG:=[dHG/S]; GF:=[dGF/NW]
GH	FG;GI;GG	FG:=[dFG/N]; GI:=[dGI/SE]; GG:=[dGG/SW]
GI	GJ;HI;GH	GJ:=[dGJ/NE]; HI:=[dHI/S]; GH:=[dGH/NW]
GJ	FI;GK;GI	FI:=[dFI/N]; GK:=[dGK/SE]; GI:=[dGI/SW]
GK	GL;HK;GJ	GL:=[dGL/NE]; HK:=[dHK/S]; GJ:=[dGJ/NW]
GL	FK;GM;GK	FK:=[dFK/N]; GM:=[dGM/SE]; GK:=[dGK/SW]
GM	GN;HM;GL	GN:=[dGN/NE]; HM:=[dHM/S]; GL:=[dGL/NW]
GN	FM;GO;GM	FM:=[dFM/N]; GO:=[dGO/SE]; GM:=[dGM/SW]
GO	GP;HO;GN	GP:=[dGP/NE]; HO:=[dHO/S]; GN:=[dGN/NW]
GP	FO;GQ;GO	FO:=[dFO/N]; GQ:=[dGQ/SE]; GO:=[dGO/SW]
GQ	HQ;GP	HQ:=[dHQ/S]; GP:=[dGP/NW]

H0	G0;H1	G0:=[dG0/N]; H1:=[dH1/SE]
H1	H2;I0;H0	H2:=[dH2/NE]; I0:=[dI0/S]; H0:=[dH0/NW]
H2	G2;H3;H1	G2:=[dG2/N]; H3:=[dH3/SE]; H1:=[dH1/SW]
H3	H4;I2;H2	H4:=[dH4/NE]; I2:=[dI2/S]; H2:=[dH2/NW]
H4	G4;H5;H3	G4:=[dG4/N]; H5:=[dH5/SE]; H3:=[dH3/SW]
H5	H6;I4;H4	H6:=[dH6/NE]; I4:=[dI4/S]; H4:=[dH4/NW]
H6	G6;H7;H5	G6:=[dG6/N]; H7:=[dH7/SE]; H5:=[dH5/SW]
H7	H8;I6;H6	H8:=[dH8/NE]; I6:=[dI6/S]; H6:=[dH6/NW]
H8	G8;H9;H7	G8:=[dG8/N]; H9:=[dH9/SE]; H7:=[dH7/SW]
H9	HA;I8;H8	HA:=[dHA/NE]; I8:=[dI8/S]; H8:=[dH8/NW]
HA	GA;HB;H9	GA:=[dGA/N]; HB:=[dHB/SE]; H9:=[dH9/SW]
HB	HC;IA;HA	HC:=[dHC/NE]; IA:=[dIA/S]; HA:=[dHA/NW]
HC	GC;HD;HB	GC:=[dGC/N]; HD:=[dHD/SE]; HB:=[dHB/SW]
HD	HE;IC;HC	HE:=[dHE/NE]; IC:=[dIC/S]; HC:=[dHC/NW]
HE	GE;HF;HD	GE:=[dGE/N]; HF:=[dHF/SE]; HD:=[dHD/SW]
HF	HG;IE;HE	HG:=[dHG/NE]; IE:=[dIE/S]; HE:=[dHE/NW]
HG	GG;HH;HF	GG:=[dGG/N]; HH:=[dHH/SE]; HF:=[dHF/SW]
HH	HI;IG;HG	HI:=[dHI/NE]; IG:=[dIG/S]; HG:=[dHG/NW]
HI	GI;HJ;HH	GI:=[dGI/N]; HJ:=[dHJ/SE]; HH:=[dHH/SW]
HJ	HK;II;HI	HK:=[dHK/NE]; II:=[dII/S]; HI:=[dHI/NW]
HK	GK;HL;HJ	GK:=[dGK/N]; HL:=[dHL/SE]; HJ:=[dHJ/SW]
HL	HM;IK;HK	HM:=[dHM/NE]; IK:=[dIK/S]; HK:=[dHK/NW]
HM	GM;HN;HL	GM:=[dGM/N]; HN:=[dHN/SE]; HL:=[dHL/SW]
HN	HO;IM;HM	HO:=[dHO/NE]; IM:=[dIM/S]; HM:=[dHM/NW]
HO	GO;HP;HN	GO:=[dGO/N]; HP:=[dHP/SE]; HN:=[dHN/SW]
HP	HQ;IO;HO	HQ:=[dHQ/NE]; IO:=[dIO/S]; HO:=[dHO/NW]
HQ	GQ;HP	GQ:=[dGQ/N]; HP:=[dHP/SW]
I0	H1;I1	H1:=[dH1/N]; I1:=[dI1/SE]
I1	I2;J0;I0	I2:=[dI2/NE]; J0:=[dJ0/S]; I0:=[dI0/NW]
I2	H3;I3;I1	H3:=[dH3/N]; I3:=[dI3/SE]; I1:=[dI1/SW]
I3	I4;J2;I2	I4:=[dI4/NE]; J2:=[dJ2/S]; I2:=[dI2/NW]
I4	H5;I5;I3	H5:=[dH5/N]; I5:=[dI5/SE]; I3:=[dI3/SW]
I5	I6;J4;I4	I6:=[dI6/NE]; J4:=[dJ4/S]; I4:=[dI4/NW]
I6	H7;I7;I5	H7:=[dH7/N]; I7:=[dI7/SE]; I5:=[dI5/SW]
I7	I8;J6;I6	I8:=[dI8/NE]; J6:=[dJ6/S]; I6:=[dI6/NW]
I8	H9;I9;I7	H9:=[dH9/N]; I9:=[dI9/SE]; I7:=[dI7/SW]
I9	IA;J8;I8	IA:=[dIA/NE]; J8:=[dJ8/S]; I8:=[dI8/NW]
IA	HB;IB;I9	HB:=[dHB/N]; IB:=[dIB/SE]; I9:=[dI9/SW]
IB	IC;JA;IA	IC:=[dIC/NE]; JA:=[dJA/S]; IA:=[dIA/NW]
IC	HD;ID;IB	HD:=[dHD/N]; ID:=[dID/SE]; IB:=[dIB/SW]
ID	IE;JC;IC	IE:=[dIE/NE]; JC:=[dJC/S]; IC:=[dIC/NW]
IE	HF;IF;ID	HF:=[dHF/N]; IF:=[dIF/SE]; ID:=[dID/SW]
IF	IG;JE;IE	IG:=[dIG/NE]; JE:=[dJE/S]; IE:=[dIE/NW]
IG	HH;IH;IF	HH:=[dHH/N]; IH:=[dIH/SE]; IF:=[dIF/SW]
IH	II;JG;IG	II:=[dII/NE]; JG:=[dJG/S]; IG:=[dIG/NW]
II	HJ;IJ;IH	HJ:=[dHJ/N]; IJ:=[dIJ/SE]; IH:=[dIH/SW]
IJ	IK;JI;II	IK:=[dIK/NE]; JI:=[dJI/S]; II:=[dII/NW]
IK	HL;IL;IJ	HL:=[dHL/N]; IL:=[dIL/SE]; IJ:=[dIJ/SW]
IL	IM;JK;IK	IM:=[dIM/NE]; JK:=[dJK/S]; IK:=[dIK/NW]
IM	HN;IN;IL	HN:=[dHN/N]; IN:=[dIN/SE]; IL:=[dIL/SW]
IN	IO;JM;IM	IO:=[dIO/NE]; JM:=[dJM/S]; IM:=[dIM/NW]
IO	HP;IN	HP:=[dHP/N]; IN:=[dIN/SW]

J0	I1;J1	I1:=[dI1/N]; J1:=[dJ1/SE]
J1	J2;K0;J0	J2:=[dJ2/NE]; K0:=[dK0/S]; J0:=[dJ0/NW]
J2	I3;J3;J1	I3:=[dI3/N]; J3:=[dJ3/SE]; J1:=[dJ1/SW]
J3	J4;K2;J2	J4:=[dJ4/NE]; K2:=[dK2/S]; J2:=[dJ2/NW]
J4	I5;J5;J3	I5:=[dI5/N]; J5:=[dJ5/SE]; J3:=[dJ3/SW]
J5	J6;K4;J4	J6:=[dJ6/NE]; K4:=[dK4/S]; J4:=[dJ4/NW]
J6	I7;J7;J5	I7:=[dI7/N]; J7:=[dJ7/SE]; J5:=[dJ5/SW]
J7	J8;K6;J6	J8:=[dJ8/NE]; K6:=[dK6/S]; J6:=[dJ6/NW]
J8	I9;J9;J7	I9:=[dI9/N]; J9:=[dJ9/SE]; J7:=[dJ7/SW]
J9	JA;K8;J8	JA:=[dJA/NE]; K8:=[dK8/S]; J8:=[dJ8/NW]
JA	IB;JB;J9	IB:=[dIB/N]; JB:=[dJB/SE]; J9:=[dJ9/SW]
JB	JC;KA;JA	JC:=[dJC/NE]; KA:=[dKA/S]; JA:=[dJA/NW]
JC	ID;JD;JB	ID:=[dID/N]; JD:=[dJD/SE]; JB:=[dJB/SW]
JD	JE;KC;JC	JE:=[dJE/NE]; KC:=[dKC/S]; JC:=[dJC/NW]
JE	IF;JF;JD	IF:=[dIF/N]; JF:=[dJF/SE]; JD:=[dJD/SW]
JF	JG;KE;JE	JG:=[dJG/NE]; KE:=[dKE/S]; JE:=[dJE/NW]
JG	IH;JH;JF	IH:=[dIH/N]; JH:=[dJH/SE]; JF:=[dJF/SW]
JH	JI;KG;JG	JI:=[dJI/NE]; KG:=[dKG/S]; JG:=[dJG/NW]
JI	IJ;JJ;JH	IJ:=[dIJ/N]; JJ:=[dJJ/SE]; JH:=[dJH/SW]
JJ	JK;KI;JI	JK:=[dJK/NE]; KI:=[dKI/S]; JI:=[dJI/NW]
JK	IL;JL;JJ	IL:=[dIL/N]; JL:=[dJL/SE]; JJ:=[dJJ/SW]
JL	JM;KK;JK	JM:=[dJM/NE]; KK:=[dKK/S]; JK:=[dJK/NW]
JM	IN;JL	IN:=[dIN/N]; JL:=[dJL/SW]
K0	J1;K1	J1:=[dJ1/N]; K1:=[dK1/SE]
K1	K2;L0;K0	K2:=[dK2/NE]; L0:=[dL0/S]; K0:=[dK0/NW]
K2	J3;K3;K1	J3:=[dJ3/N]; K3:=[dK3/SE]; K1:=[dK1/SW]
K3	K4;L2;K2	K4:=[dK4/NE]; L2:=[dL2/S]; K2:=[dK2/NW]
K4	J5;K5;K3	J5:=[dJ5/N]; K5:=[dK5/SE]; K3:=[dK3/SW]
K5	K6;L4;K4	K6:=[dK6/NE]; L4:=[dL4/S]; K4:=[dK4/NW]
K6	J7;K7;K5	J7:=[dJ7/N]; K7:=[dK7/SE]; K5:=[dK5/SW]
K7	K8;L6;K6	K8:=[dK8/NE]; L6:=[dL6/S]; K6:=[dK6/NW]
K8	J9;K9;K7	J9:=[dJ9/N]; K9:=[dK9/SE]; K7:=[dK7/SW]
K9	KA;L8;K8	KA:=[dKA/NE]; L8:=[dL8/S]; K8:=[dK8/NW]
KA	JB;KB;K9	JB:=[dJB/N]; KB:=[dKB/SE]; K9:=[dK9/SW]
KB	KC;LA;KA	KC:=[dKC/NE]; LA:=[dLA/S]; KA:=[dKA/NW]
KC	JD;KD;KB	JD:=[dJD/N]; KD:=[dKD/SE]; KB:=[dKB/SW]
KD	KE;LC;KC	KE:=[dKE/NE]; LC:=[dLC/S]; KC:=[dKC/NW]
KE	JF;KF;KD	JF:=[dJF/N]; KF:=[dKF/SE]; KD:=[dKD/SW]
KF	KG;LE;KE	KG:=[dKG/NE]; LE:=[dLE/S]; KE:=[dKE/NW]
KG	JH;KH;KF	JH:=[dJH/N]; KH:=[dKH/SE]; KF:=[dKF/SW]
KH	KI;LG;KG	KI:=[dKI/NE]; LG:=[dLG/S]; KG:=[dKG/NW]
KI	JJ;KJ;KH	JJ:=[dJJ/N]; KJ:=[dKJ/SE]; KH:=[dKH/SW]
KJ	KK;LI;KI	KK:=[dKK/NE]; LI:=[dLI/S]; KI:=[dKI/NW]
KK	JL;KJ	JL:=[dJL/N]; KJ:=[dKJ/SW]

L0	K1;L1	K1:=[dK1/N]; L1:=[dL1/SE]
L1	L2;M0;L0	L2:=[dL2/NE]; M0:=[dM0/S]; L0:=[dL0/NW]
L2	K3;L3;L1	K3:=[dK3/N]; L3:=[dL3/SE]; L1:=[dL1/SW]
L3	L4;M2;L2	L4:=[dL4/NE]; M2:=[dM2/S]; L2:=[dL2/NW]
L4	K5;L5;L3	K5:=[dK5/N]; L5:=[dL5/SE]; L3:=[dL3/SW]
L5	L6;M4;L4	L6:=[dL6/NE]; M4:=[dM4/S]; L4:=[dL4/NW]
L6	K7;L7;L5	K7:=[dK7/N]; L7:=[dL7/SE]; L5:=[dL5/SW]
L7	L8;M6;L6	L8:=[dL8/NE]; M6:=[dM6/S]; L6:=[dL6/NW]
L8	K9;L9;L7	K9:=[dK9/N]; L9:=[dL9/SE]; L7:=[dL7/SW]
L9	LA;M8;L8	LA:=[dLA/NE]; M8:=[dM8/S]; L8:=[dL8/NW]
LA	KB;LB;L9	KB:=[dKB/N]; LB:=[dLB/SE]; L9:=[dL9/SW]
LB	LC;MA;LA	LC:=[dLC/NE]; MA:=[dMA/S]; LA:=[dLA/NW]
LC	KD;LD;LB	KD:=[dKD/N]; LD:=[dLD/SE]; LB:=[dLB/SW]
LD	LE;MC;LC	LE:=[dLE/NE]; MC:=[dMC/S]; LC:=[dLC/NW]
LE	KF;LF;LD	KF:=[dKF/N]; LF:=[dLF/SE]; LD:=[dLD/SW]
LF	LG;ME;LE	LG:=[dLG/NE]; ME:=[dME/S]; LE:=[dLE/NW]
LG	KH;LH;LF	KH:=[dKH/N]; LH:=[dLH/SE]; LF:=[dLF/SW]
LH	LI;MG;LG	LI:=[dLI/NE]; MG:=[dMG/S]; LG:=[dLG/NW]
LI	KJ;LH	KJ:=[dKJ/N]; LH:=[dLH/SW]
M0	L1;M1	L1:=[dL1/N]; M1:=[dM1/SE]
M1	M2;N0;M0	M2:=[dM2/NE]; N0:=[dN0/S]; M0:=[dM0/NW]
M2	L3;M3;M1	L3:=[dL3/N]; M3:=[dM3/SE]; M1:=[dM1/SW]
M3	M4;N1;M2	M4:=[dM4/NE]; N2:=[dN2/S]; M2:=[dM2/NW]
M4	L5;M5;M3	L5:=[dL5/N]; M5:=[dM5/SE]; M3:=[dM3/SW]
M5	M6;N2;M4	M6:=[dM6/NE]; N2:=[dN2/S]; M4:=[dM4/NW]
M6	L7;M7;M5	L7:=[dL7/N]; M7:=[dM7/SE]; M5:=[dM5/SW]
M7	M8;N3;M6	M8:=[dM8/NE]; N3:=[dN3/S]; M6:=[dM6/NW]
M8	L9;M9;M7	L9:=[dL9/N]; M9:=[dM9/SE]; M7:=[dM7/SW]
M9	MA;N4;M8	MA:=[dMA/NE]; N4:=[dN4/S]; M8:=[dM8/NW]
MA	LB;MB;M9	LB:=[dLB/N]; MB:=[dMB/SE]; M9:=[dM9/SW]
MB	MC;N5;MA	MC:=[dMC/NE]; N5:=[dN5/S]; MA:=[dMA/NW]
MC	LD;MD;MB	LD:=[dLD/N]; MD:=[dMD/SE]; MB:=[dMB/SW]
MD	ME;N6;MC	ME:=[dME/NE]; N6:=[dN6/S]; MC:=[dMC/NW]
ME	LF;MF;MD	LF:=[dLF/N]; MF:=[dMF/SE]; MD:=[dMD/SW]
MF	MG;N7;ME	MG:=[dMG/NE]; N7:=[dN7/S]; ME:=[dME/NW]
MG	LH;MF	LH:=[dLH/N]; MF:=[dMF/SW]
N0	M1	M1:=[dM1/N]
N1	M3	M3:=[dM3/N]
N2	M5	M5:=[dM5/N]
N3	M7	M7:=[dM7/N]
N4	M9	M9:=[dM9/N]
N5	MB	MB:=[dMB/N]
N6	MD	MD:=[dMD/N]
N7	MF	MF:=[dMF/N]
HEXAGON MATRIX TABLE NODES' PROPERTY		

Table-12: HEXAGON NEIGHBOR-TO-NEIGHBORS NODES' PROPERTIES TABLE

G-ROUTING ALGORITHM METHODOLOGY

Henry V. Pham

START SOURCE NODE 'C5'				START DESTINATION NODE 'K9'		
Steps	Node	Neighbor Nodes		Steps	Node	Neighbor Nodes
1	C5	B4, C6, C4		1	K9	KA, L8, K8
1.1	B4	B5, B3		1.1	KA	JB, KB
1.2	C6	C7, D7		1.2	L8	L9, L7
1.3	C4	D5, C3		1.3	K8	J9, K7
1.1.1	B5	A2		1.1.1	JB	JC
1.1.2	B3	A1		1.1.2	KB	KC
1.2.1	C7	C8		1.2.1	L9	M8
1.2.2	D7	D8		1.2.2	L7	M6
1.3.1	D5	D4		1.3.1	J9	J8
1.3.2	C3	C2		1.3.2	K7	K6
1.1.1.1	A2	xxx		1.1.1.1	JC	ID
1.1.2.1	A1	xxx		1.1.2.1	KC	KD
1.2.1.1	C8	C9		1.2.1.1	M8	M9
1.2.2.1	D8	E9		1.2.2.1	M6	M5
1.3.1.1	D4	E5		1.3.1.1	J8	I9
1.3.2.1	C2	C1		1.3.2.1	K6	K5
1.2.1.1.1	C9	CA		1.1.1.1.1	ID	IE
1.2.2.1.1	E9	EA		1.1.2.1.1	KD	KE
1.3.1.1.1	E5	E4		1.2.1.1.1	M9	N4
1.3.2.1.1	C1	C0		1.2.2.1.1	M5	N2
1.2.1.1.1.1	CA	CB		1.3.1.1.1	I9	I8
1.2.2.1.1.1	EA	FB		1.3.2.1.1	K5	K4
1.3.1.1.1.1	E4	F5		1.1.1.1.1.1	IE	HF
1.3.2.1.1.1	C0	xxx		1.1.2.1.1.1	KE	KF
1.2.1.1.1.1.1	CB	CC		1.2.1.1.1.1	N4	xxx
1.2.2.1.1.1.1	FB	FC		1.2.2.1.1.1	N2	xxx
1.3.1.1.1.1.1	F5	F4		1.3.1.1.1.1	I8	H9
1.2.1.1.1.1.1.1	CC	CD		1.3.2.1.1.1	K4	K3
1.2.2.1.1.1.1.1	FC	GD		1.1.1.1.1.1.1	HF	HG
1.3.1.1.1.1.1.1	F4	G5		1.1.2.1.1.1.1	KF	KG
1.2.1.1.1.1.1.1.1	CD	CE		1.3.1.1.1.1.1	H9	H8
1.2.2.1.1.1.1.1.1	GD	GE		1.3.2.1.1.1.1	K3	K2
1.3.1.1.1.1.1.1.1	G5	G4		1.1.1.1.1.1.1.1	HG	GG
1.2.1.1.1.1.1.1.1.1	CE	CF		1.1.2.1.1.1.1.1	KG	KH
1.2.2.1.1.1.1.1.1.1	GE	HE		1.3.1.1.1.1.1.1	H8	G8
1.3.1.1.1.1.1.1.1.1	G4	H4		1.3.2.1.1.1.1.1	K2	K1
1.2.1.1.1.1.1.1.1.1.1	CF	CG		1.1.1.1.1.1.1.1.1	GG	GH
1.2.2.1.1.1.1.1.1.1.1	HE	HF (Crossed)		1.1.2.1.1.1.1.1.1	KH	KI
1.3.1.1.1.1.1.1.1.1.1	H4	H3		1.3.1.1.1.1.1.1.1	G8	G7
1.2.1.1.1.1.1.1.1.1.1.1	CG	CH		1.3.2.1.1.1.1.1.1	K1	K0
1.2.2.1.1.1.1.1.1.1.1.1	HF	IE		1.1.1.1.1.1.1.1.1.1	GH	FG
1.3.1.1.1.1.1.1.1.1.1.1	H3	I2		1.1.2.1.1.1.1.1.1.1	KI	KJ
1.2.1.1.1.1.1.1.1.1.1.1.1	CH	CI		1.3.1.1.1.1.1.1.1.1	G7	F6
1.2.2.1.1.1.1.1.1.1.1.1.1	IE	IF		1.3.2.1.1.1.1.1.1.1	K0	xxx
1.3.1.1.1.1.1.1.1.1.1.1.1	I2	I1		1.1.1.1.1.1.1.1.1.1.1	FG	FH
1.2.2.1.1.1.1.1.1.1.1.1.1.1	IF	JE		1.1.2.1.1.1.1.1.1.1.1	KJ	KK
1.3.1.1.1.1.1.1.1.1.1.1.1.1	I1	J0		1.3.1.1.1.1.1.1.1.1.1	F6	F5 (Crossed)
HEXAGON SEARCH CROSSING PARALLELOGRAM SECTION IN 30° DIRECTIONS WITH G-ROUTING SEQUENCES						

Table-13: FAST SEARCH 30° HEXAGON OUTWARD DIRECTIONS G-ROUTING ALGORITHM

henryvpham@gmail.com

Hexagon Optimization G-ROUTING ALGORITHM can be quickly search for crossing section from source and destination nodes by starting with clockwise in outward direction with **30° turns** at 12 O' clock in directions of lines '**C5.SW LINE**', '**C5.EW LINE**', and '**C5.SE LINE**' at node '**C5**'; and in directions of lines '**K9.NE LINE**', '**K9.EW LINE**', and '**K9.SE LINE**' at node '**K9**'. The sequences below show step-by-step for the optimization routing for the source node '**C5**' to destination node '**K9**'. **Table-13** shows sequences of step-by-step for quick searching for routes along the crossed parallelogram section.

1. Start with source node '**C5**' on the **Left-Side** and based on the database **Table-12**, there are 3 neighbor nodes '**B4, C6, C4**' in order of clockwise direction start at 12 O' clock; the following step '1.1' starts with 1st node '**B4**' with the outward 30° directions has neighbor nodes '**B5, B3**'; the next step '1.2' starts with 2nd node '**C6**' with the outward 30° directions has neighbor nodes '**C7, D7**'; the next steps would be the same for the other nodes '**C4**' with the neighbor nodes '**D5, C3**' in step '1.3', and '**B5**' with neighbor node '**A2**' in step '1.1.1'.
2. Start with destination node '**K9'** on the **Right-Side**, there are 3 neighbor nodes '**KA, L8, K8**' in order of clockwise direction start at 12 O' clock; the following step '1.1' starts with 1st node '**KA**' with the outward 30° directions has neighbor nodes '**JB, KB**'; the next step '1.2' starts with 2nd node '**L8**' with outward 30° directions has neighbor nodes '**L9, L7**'; the next steps would be the same for the other nodes '**K8**' with the neighbor nodes '**J9, K7**' in step '1.3' and '**JB**' with neighbor node '**JC**' in steps '1.1.1'.
3. Continue from both sides step-by-step with the sequences shown in **Table-13**, the optimization algorithm found shared node '**HF**' in South-East direction of the source node '**C5**' and in North-East direction of destination node '**K9**'; found another shared node '**F5**' in South-West direction of the source node '**C5**' and in North-West direction of destination node '**K9**'. These 2 shared nodes will create the 2 routes from the source to the destination node.
4. With this quick routing methodology, the G-ROUTING algorithm found the crossed parallelogram with 2 fastest routes around this parallelogram. **Table-14** shows these 2 routes with the same number of 12 routing nodes and the same number of turns.

This optimization is perfect routing method for the uniform network layout; hexagon network layout can be used in future for cells structure network layout.

5. **Figure-8** below shows graphical layout and directions of this optimization G-ROUTING ALGORITHM for the hexagon uniform network layout.

ROUTE STEPS	NODE	DIRECTION	DISTANCE	ROUTE STEPS	NODE	DIRECTION	DISTANCE
1/1	C5	START	0.00	2/1	C5	START	0.00
1/2	C6	SE	dC6	2/2	C4	SW	dC4
1/3	D7	S	dD7	2/3	D5	S	dD5
1/4	D8	SE	dD8	2/4	D4	SW	dD4
1/5	E9	S	dE9	2/5	E5	S	dE5
1/6	EA	SE	dEA	2/6	E4	SW	dE4
1/7	FB	S	dFB	2/7	F5	S	dF5
1/8	FC	SE	dFC	2/8	F6	SE	dF6
1/9	GD	S	dGD	2/9	G7	S	dG7
1/10	GE	SE	dGE	2/10	G8	SE	dG8
1/11	HE	S	dHE	2/11	H8	S	dHC
1/12	HF	SE	dHF	2/12	H9	SE	dHD
1/13	IE	S	dIE	2/13	I8	S	dIC
1/14	ID	SW	dID	2/14	I9	SE	dID
1/15	JC	S	dJC	2/15	J8	S	dJC
1/16	JB	SW	dJB	2/16	J9	SE	dJB
1/17	KA	S	dKA	2/17	K8	S	dKA
1/18	K9	SW	dK9	2/18	K9	SE	dK9
Steps := 18	No. Turns:=1 (SE/S,S/SW)		Total Distance	Steps := 18	No. Turns:=1 (SW/S,S/SE)		Total Distance

Table-14: FOUND 2 ROUTES W/ OPTIMIZATION HEXAGON G-ROUTING ALGORITHM

To complete the routing within the crossed parallelogram section to find more possible faster routes if the network has shortcut nodes. **Table-15** below shows the sequences of routing step-by-step to search for any possible faster routes within the crossed section.

1. Start with source node '**C5**' on the **Left-Side** and nodes' properties from **Table-12**, there are 2 neighbor nodes '**C6, C4**' in order of clockwise direction start at 12 O' clock within the crossed section and within 30° angle **South-East** and **South-West** (SE/SW) directions; the following step '1.1' starts with 1st node '**C6**' with 30° SE/SW neighbor nodes '**D7**'; the next step '1.2' starts with 2nd node '**C4**' with 30° SE/SW neighbor nodes '**D5**'; the next steps would be the same for the other nodes '**D7**' with the neighbor nodes '**D8, D6**' and node '**D5**' with the neighbor nodes '**D6, D4**' with duplicated node '**D6**' from the previous steps in steps '1.1.1' and '1.2.1' for nodes '**D7**' and '**D5**'.
2. Start with destination node '**K9'** on the **Right-Side**, there are 2 neighbor nodes '**KA, K8**' in order of clockwise direction start at 12 O' clock within the crossed section and within 30° angle **North-East** and **North-West** (NE/NW) directions; the following step '1.1' starts with 1st node '**KA**' with 30° NE/NW neighbor node '**JB**'; the next step '1.2' starts with 2nd node '**K8**' with 30° NE/NW neighbor node '**J9**'; the next steps would be the same for the other nodes '**JB**' with the neighbor nodes '**JC, JA**' and '**J9**' with the neighbor nodes '**JA, J8**' with duplicated node '**JA**' in steps '1.1.1' and '1.2.1'.
3. Continue from both sides step-by-step with the sequences shown in **Table-15,** searching for the neighbor nodes within the crossed section from the source node '**C5**' to the destination node '**K9**'. As shown in **Table-15** the algorithm found the 1st shared node '**GD**' at step '1.1.1.1.1.1.1.1' in direction from the source node '**C5**' and at step '1.1.1.1.1.1.1.1.1' in direction from the destination node '**K9**'; found the 2st shared node '**GC**' at step '1.1.1.1.1.1.1.1.1' in direction from the source node '**C5**' and at step '1.1.1.1.1.2.1.2' in direction from the destination node '**K9**'; and found the 3st shared node '**GA**' at step '1.1.1.1.1.1.1.2.2' in direction from the source node '**C5**' and at step '1.1.1.2.1.2.1.2' in direction from the destination node '**K9**'. These 3 shared nodes will create the 3 routes from the source to the destination node; given 3 routes are good enough to compare and to choose the best one.
4. Finally, links these shared nodes with their paths together for these 3 routes as shown in **Table-16** below with the best route with series of routing nodes '**C5**, **C6, D7, D8, E9, EA, FB, FC, GD, GE, HE, HF, IE, ID, JC, JB, KA, LB, K9**' with 1 turn. **Figure-8** shows graphical directions for all 3 routes above within the crossed parallelogram section.

henryvpham@gmail.com

G-ROUTING ALGORITHM METHODOLOGY

Henry V. Pham

START SOURCE NODE 'C5'				START DESTINATION NODE 'K9'		
Steps	Node	Neighbor Nodes		Steps	Node	Neighbor Nodes
1	C5	C6, C4		1	K9	KA, K8
1.1	C6	D7		1.1	KA	JB
1.2	C4	D5		1.2	K8	J9
1.1.1	D7	D8, D6		1.1.1	JB	JC, JA
1.2.1	D5	D6, D4		1.2.1	J9	JA, J8
1.1.1.1	D8	E9		1.1.1.1	JC	ID
1.1.1.2	D6	D7, E7		1.1.1.2	JA	IB, J9
1.2.1.2	D4	D5, E5		1.2.1.2	J8	I9
1.1.1.1.1	E9	EA, E8		1.1.1.1.1	ID	IE, IC
1.1.1.2.2	E7	E8, E6		1.1.1.2.1	IB	IC, IA
1.2.1.2.2	E5	E6, E4		1.2.1.2.1	I9	IA, I8
1.1.1.1.1.1	EA	FB		1.1.1.1.1.1	IE	HF, ID
1.1.1.1.1.2	E8	E9, F9		1.1.1.1.1.2	IC	HD, IB
1.1.1.2.2.2	E6	E7, F7		1.1.1.2.1.2	IA	HB, I9
1.2.1.2.2.2	E4	E5, F5		1.2.1.2.1.2	I8	H9
1.1.1.1.1.1.1	FB	FC, FA		1.1.1.1.1.1.1	HF	HE
1.1.1.1.1.2.2	F9	FA, F8		1.1.1.1.1.2.1	HD	HE, HC
1.1.1.2.2.2.2	F7	F8, F6		1.1.1.2.1.2.1	HB	HC, HA
1.2.1.2.2.2.2	F5	F6		1.2.1.2.1.2.1	H9	HA, H8
1.1.1.1.1.1.1.1	FC	GD		1.1.1.1.1.1.1.1	HE	GE, HD
1.1.1.1.1.1.1.2	FA	FB, GB		1.1.1.1.1.2.1.2	HC	GC, HB
1.1.1.1.1.2.2.2	F8	F9, G9		1.1.1.2.1.2.1.2	HA	GA, H9
1.1.1.2.2.2.2.2	F6	F7, G7		1.2.1.2.1.2.1.2	H8	G8
1.1.1.1.1.1.1.1.1	GD	GE, GC		1.1.1.1.1.1.1.1.1	GE	GD
1.1.1.1.1.1.1.2.2	GB	GC, GA		1.1.1.1.1.2.1.2.1	GC	GD, GB
1.1.1.1.1.2.2.2.2	G9	GA, G8		1.1.1.2.1.2.1.2.1	GA	GB, G9
HEXAGON OPTIMIZATION WITHIN CROSSED PARALLELOGRAM G-ROUTING SEQUENCES						

Table-15: SEARCH WITHIN CROSSED PARALLELOGRAM ROUTING ALGORITHM

ROUTE STEPS	NODE	DIRECTION	DISTANCE
1/1	C5	START	0.00
1/2	C6	SE	dC6
1/3	D7	S	dD7
1/4	D8	SE	dD8
1/5	E9	S	dE9
1/6	EA	SE	dEA
1/7	FB	S	dFB
1/8	FC	SE	dFC
1/9	GD	S	dGD
1/10	GE	SE	dGE
1/11	HE	S	dHE
1/12	HF	SE	dHF
1/13	IE	S	dIE
1/14	ID	SW	dID
1/15	JC	S	dJC
1/16	JB	SW	dJB
1/17	KA	S	dKA
1/18	K9	SW	dK9
Steps := 18	No. Turns:=1 (S,SW)		Total Distance

ROUTE STEPS	NODE	DIRECTION	DISTANCE
2/1	C5	START	0.00
2/2	C6	SE	dC6
2/3	D7	S	dD7
2/4	D8	SE	dD8
2/5	E9	S	dE9
2/6	EA	SE	dEA
2/7	FB	S	dFB
2/8	FC	SE	dFC
2/9	GD	S	dGD
2/10	GC	SW	dGC
2/11	HC	S	dHC
2/12	HD	SE	dHD
2/13	IC	S	dIC
2/14	ID	SE	dID
2/15	JC	S	dJC
2/16	JB	SW	dJB
2/17	KA	S	dKA
2/18	K9	SW	dK9
Steps := 18	No. Turns:=3 (S,SW,SE,SW)		Total Distance

ROUTE STEPS	NODE	DIRECTION	DISTANCE
3/1	C5	START	0.00
3/2	C6	SE	dC6
3/3	D7	S	dD7
3/4	D8	SE	dD8
3/5	E9	S	dE9
3/6	EA	SE	dEA
3/7	FB	S	dFB
3/8	FA	SW	dFA
3/9	GB	S	dGB
3/10	GA	SW	dGA
3/11	HA	S	dHA
3/12	HB	SE	dHB
3/13	IA	S	dIA
3/14	IB	SE	dIB
3/15	JA	S	dJA
3/16	JB	SE	dJB
3/17	KA	S	dKA
3/18	K9	SW	dK9
Steps := 18	No. Turns:=3 (S,SW,SE,SW)		Total Distance

Table-16: FOUND 3 ROUTES W/ OPTIMIZATION HEXAGON G-ROUTING ALGORITHM

henryvpham@gmail.com

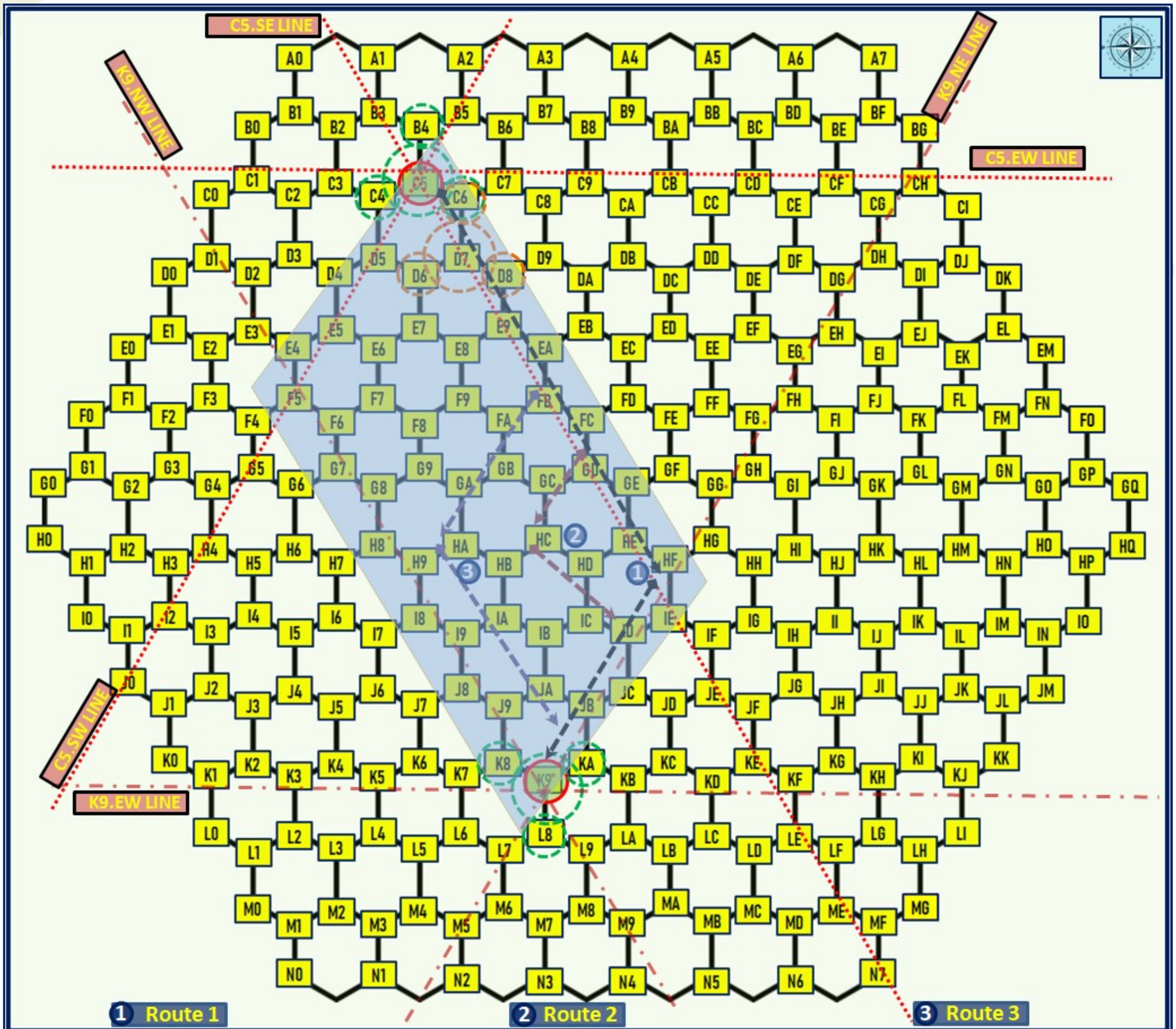
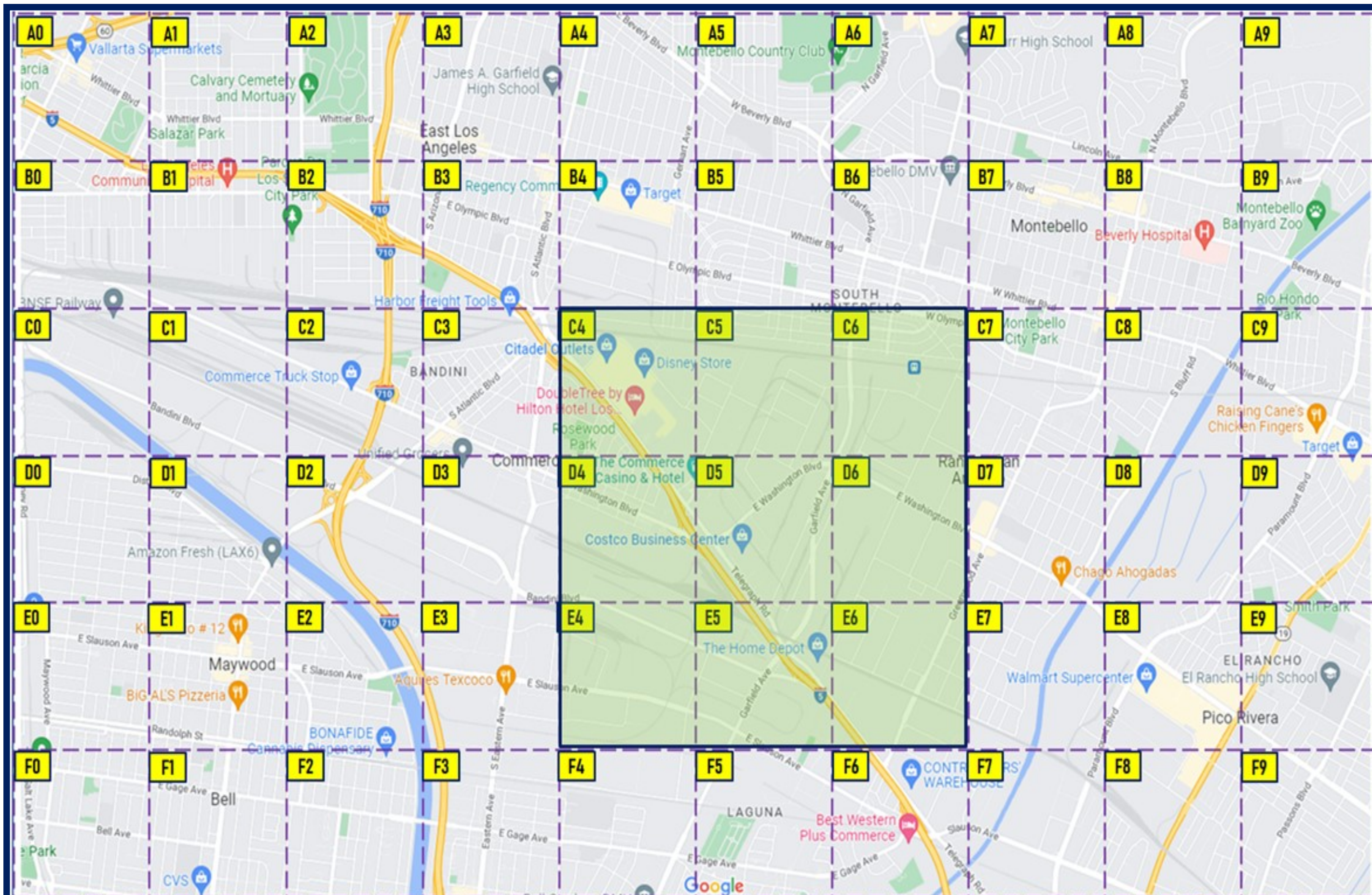


Figure-8: FOUND 3 ROUTES W/ OPTIMIZATION HEXAGON G-ROUTING ALGORITHM

Henry V. Pham

G-ROUTING ALGORITHM for other Use Cases/Applications

The G-ROUTING ALGORITHM is also useful routing algorithm for other use cases like Cell ëMap Layout in the '**Cell ëMap Live Updates System**' invention and internet networking or **3D ëWeb** network layout for '**Cloud OS – Operating System**' invention.



Cell ëMap ID	Cell ëMap Properties (Shared FWY & Streets)							Neighbors Cell ëMaps
	FWY	Street-1	Street-2	Street-3	Street-4	Street-5	Street-6	
C4	FWY-5			Telegraph Rd				B4,B5,C5,D5,D4,D3,C3,B3
C5								B5,B6,C6,D6,D5,D4,C4,B4
C6		Garfield Ave						B6,B7,C7,D7,D6,D5,C5,B5
D4	FWY-5		Washington Blvd	Telegraph Rd		Bandini Blvd		C4,C5,D5,E5,E4,E3,D3,C3
D5	FWY-5	Garfield Ave	Washington Blvd	Telegraph Rd				C5,C6,D6,E6,E5,E4,D4,C4
D6			Washington Blvd				Greenwood Ave	C6,C7,D7,E7,E6,E5,D5,C5
E4					Slauson Ave	Bandini Blvd		D4,D5,E5,F5,F4,F3,E3,D3
E5	FWY-5	Garfield Ave		Telegraph Rd	Slauson Ave	Bandini Blvd		D5,D6,E6,F6,F5,F4,E4,D4
E6	FWY-5			Telegraph Rd		Bandini Blvd	Greenwood Ave	D6,D7,E7,F7,F6,F5,E5,D5

NOTE: Neighbor-to-Neighbors Cell eMaps -- Navigation Route will have the same shared street(s) between cell neighobors

Figure-9 CELL ëMAP SHARED STREETS PROPERTIES FOR G-ROUGING ALGORITHM

Firgure-9 shows the Cell ëMap layout in highest resolution of 1km cell; each cell links to its neighbor cells with properties of connected Freeways or Streets as the neighbor nodes for routing from one location to another. **Figure-10** shows a local community StarTree ëWeb nodes layout; the star tree network is one-to-five nodes layout. One node can have 5 neighbor nodes and 1 local community node with 1 linkable community-to-community node connect on demand. This Star ëWeb nodes layout is manageable in form of group geometry layout, and this network layout can be auto drawing when a node or branch is newly connected to or removed from the network using **Neighbor-to-Neighbors** methodology which has been mentioned in the '**Cloud OS –Operating System**' invention. This is the promise internet or **Multi-layer ëWeb networks** for future networking that is the plan for Cloud OS network layout.

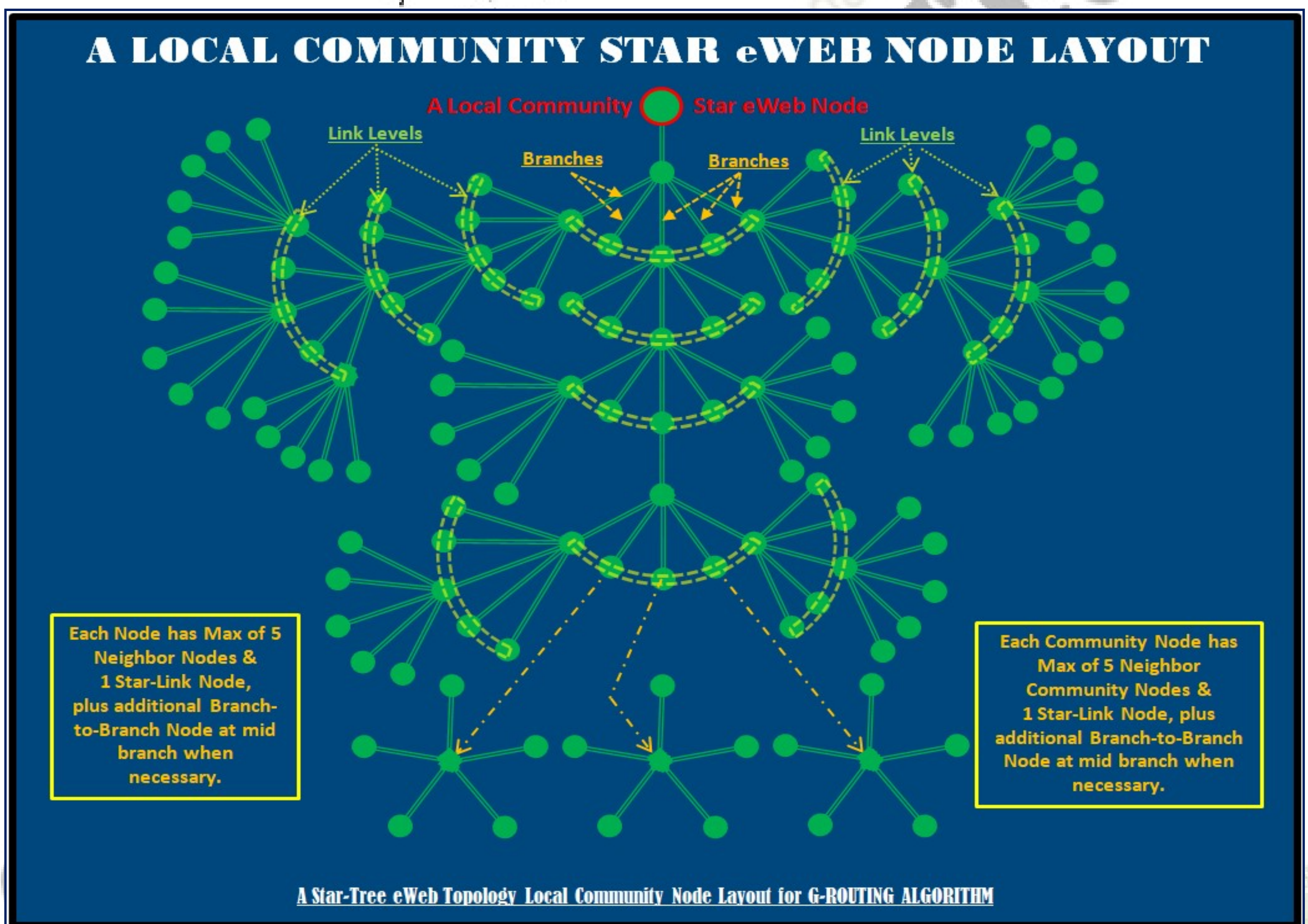


Figure-10: CLOUD OS STAR-TREE ëWEB NODES LAYOUT FOR G-ROUGING ALGORITHM

G-ROUTING ALGORITHM TX/RX Protocol

Once we have the routes from the G-ROUTING ALGORITHM, we need away to transfer data or communicate to a destination node with the G-NETWORK layout which must be in some geometry layouts that can be controllable and manageable. **Figure-11** shows the TX **Protocol Format** when a route is defined with the series of routing nodes 'N(0)…N(c)…N(n)'; where N(0) is the original source node, N(c) is the current node, and N(n) is the destination node. The node with fields that highlight in **light-green** 'N(c+1)' is the next node to be transmitted to. When the field 'N(x)-SET' is set, 'N(x)-PROP' will be filled in either by the system or will ask to be filled in when a node is holding the package to be routed to others; if this field 'N(x)-SET' is not set or '0x00' value, then the 'N(x)-PROP' will be NULL or empty. This protocol also allows the current node to adjust and re-route if the next routing neighbor node is suddenly disconnected or unavailable. Note that this node must be a re-routable node which has the database of entire network nodes. This protocol allows to set re-routing at a certain percentage 'ALLOW-REROUTE-AT-%PATH' of the routing path for optimization purposes. This dynamic protocol is useful and allows the re-routable nodes or destination node to make decision when receiving data package. **Table-17** below shows the short description of each field of the **TX Protocol Format**.

There are 3 options of response to the TX Protocol. **Figure-12** shows response **option-1** with the requirement of every receiving node must replies to the system or the source node when received the data package as an acknowledgement of receiving data package and confirmed transmitting to the next node. **Figure-13** shows response **option-2** with the requirement of only receiving node replies to the previous node when received the data package as an acknowledgment of receiving data package and confirmed transmitting to the next node; a neighbor-to-neighbor confirmation only option, this option is also known as lazy-responding option. **Figure-14** shows response **option-3** with the requirement of only the last or the destination node replies to the system or the source node when received the data package as an acknowledgement of receiving data package. This option requires a list of routing nodes to be one-by-one passing forward the acknowledgement back to the system with optional of status or statistic of the destination node or routing nodes.

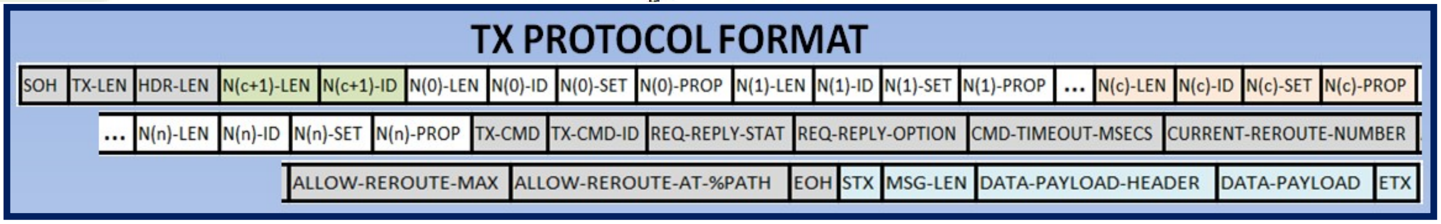


Figure-11: TX PROTOCOL FORMAT

TX PROTOCOL DESCRIPTION		
Name	**Description**	**# Bytes**
SOH	Start of Header tag	1
TX-LEN	Total Transmision Length	4*
HDR-LEN	Total Header Length	2*
N(c+1)-LEN	Next routing node ID Length (Current node +1)	<var>
N(c+1)-ID	Next routing node ID	<var>
N(0)-LEN	Source node ID Length	1
N(0)-ID	Source node ID	<var>
N(0)-SET	Source node SET attribute byte (Request or prefill node's property)	1
N(0)-PROP	If N(0)-SET defined, the node's property will be filled in along with property length	<var>
. . .	Other routing nodes... [N(c-)-LEN;N(c-)-ID;N(c-)-SET;N(c-)-PROP]	<var>
N(c)-LEN	Current node ID Length	1
N(c)-ID	Current node ID	<var>
N(c)-SET	Current node SET attribute byte (Request or prefill node's property)	1
N(c)-PROP	If N(c)-SET defined, the node's property will be filled in along with property length	<var>
. . .	Other routing nodes... [N(c+)-LEN;N(c+)-ID;N(c+)-SET;N(c+)-PROP]	<var>
N(n)-LEN	Destination node ID Length	1
N(n)-ID	Destination node ID	<var>
N(n)-SET	Destination node SET attribute byte (Request or prefill node's property)	1
N(n)-PROP	If N(n)-SET defined, the node's property will be filled in along with property length	<var>
TX-CMD	Transmit Command	1*
TX-CMD-ID	Transmit Command ID - Unique number to identify when responding	1*
REQ-REPLY-STAT	Request REPLY with node's statistic or attributes	1
REQ-REPLY-OPTION	Request REPLY option (1, 2, 3, FF)	1
CMD-TIMEOUT-MSECS	TX Command timeout (milliseconds)	4
CURRENT-REROUTE-NUMBER	Current number of reroute	1
ALLOW-REROUTE-MAX	Maximum number of reroute allow	1
ALLOW-REROUTE-AT-%PATH	Percentage (%) routing path allow to reroute (total routing nodes := 100%)	1
EOH	End of Header tag	1
STX	Start of message tag	1
MSG-LEN	Message Length	4*
DATA-PAYLOAD-HEADER	Message Data Payload Header	<var>
DATA-PAYLOAD	Message Data Payload (User Data)	<var>
ETX	End of message tag	1
NOTE: Byte width (*) can be adjust to fit the application for simple or more complex system		

Table-17: TX PROTOCOL FORMAT DESCRIPTION

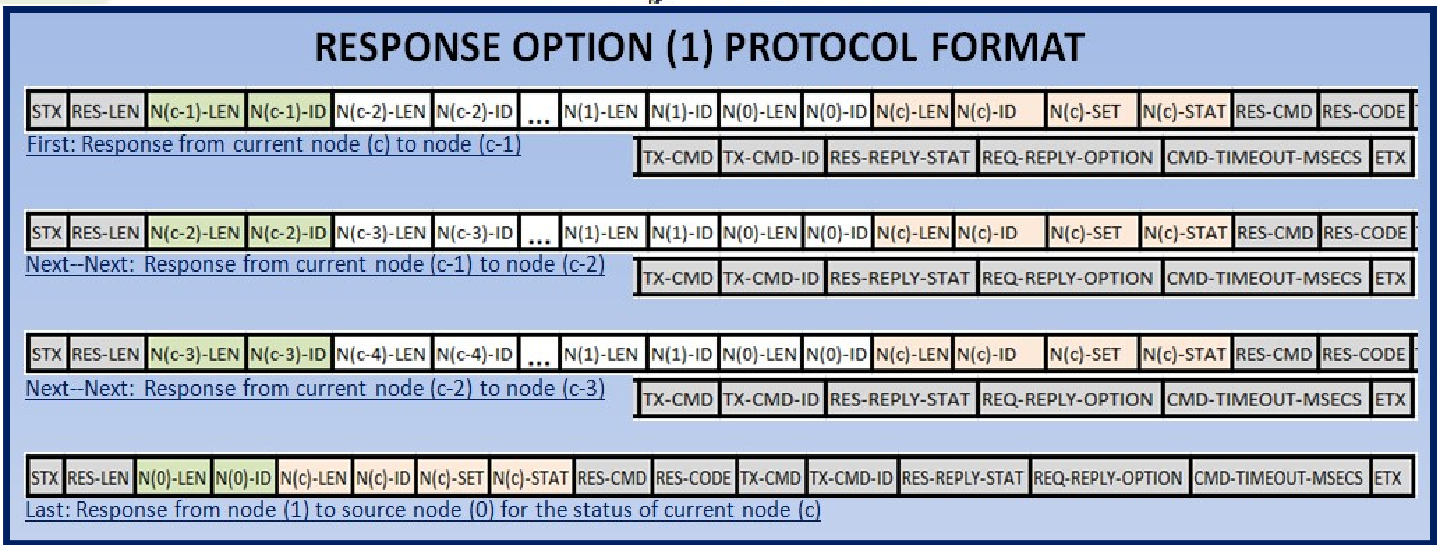


Figure-12: RX RESPONSE OPTION (1) PROTOCOL FORMAT

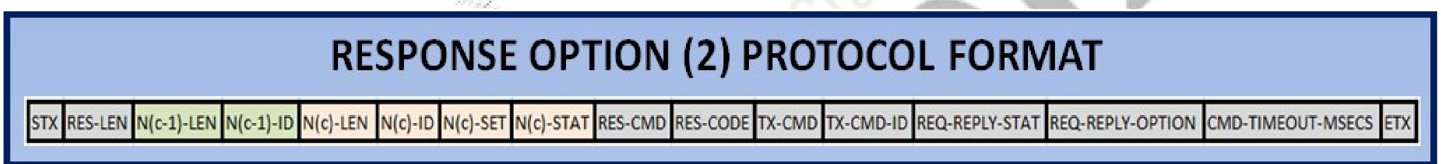


Figure-13: RX RESPONSE OPTION (2) PROTOCOL FORMAT

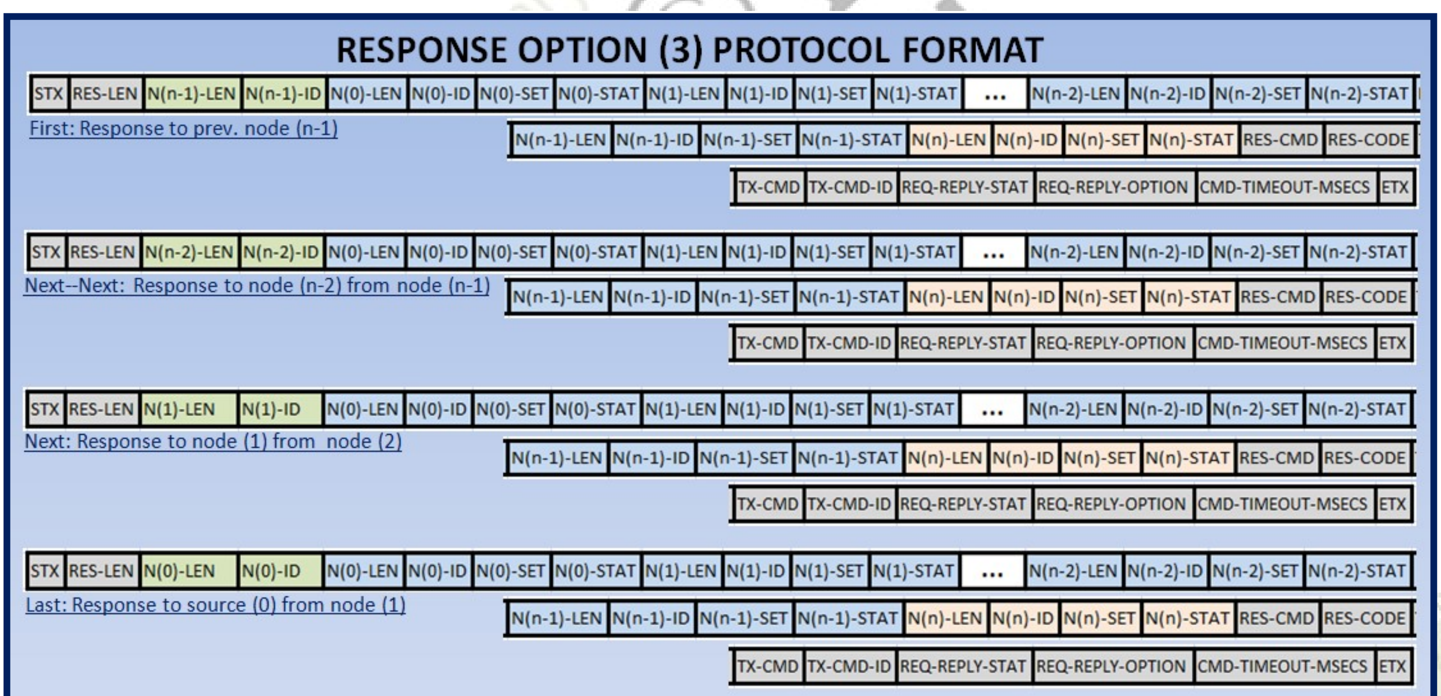


Figure-14: RX RESPONSE OPTION (3) PROTOCOL FORMAT

The protocol also supports auto notify when new nodes are added into the network or alert when a node is removed from the network. **Figure-15** shows protocol format for notifying when a node is added, removed or gone missing for some reasons with an option of broadcasting using Neighbor-to-Neighbors methodology. **Figure-16** below also shows protocol supporting of system updating with system update data package for new configuration, firmware updates or anything related to the system updates or system properties changes. **Figure-17** shows the format of node's property for status or statistic when TX protocol requests.

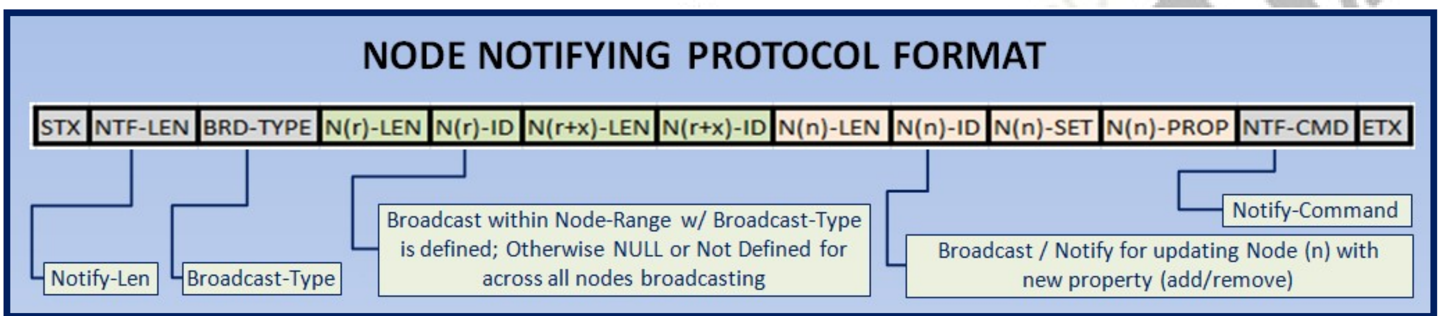


Figure-15: NOTIFYING (ADD/REMOVE/MISSING) PROTOCOL FORMAT

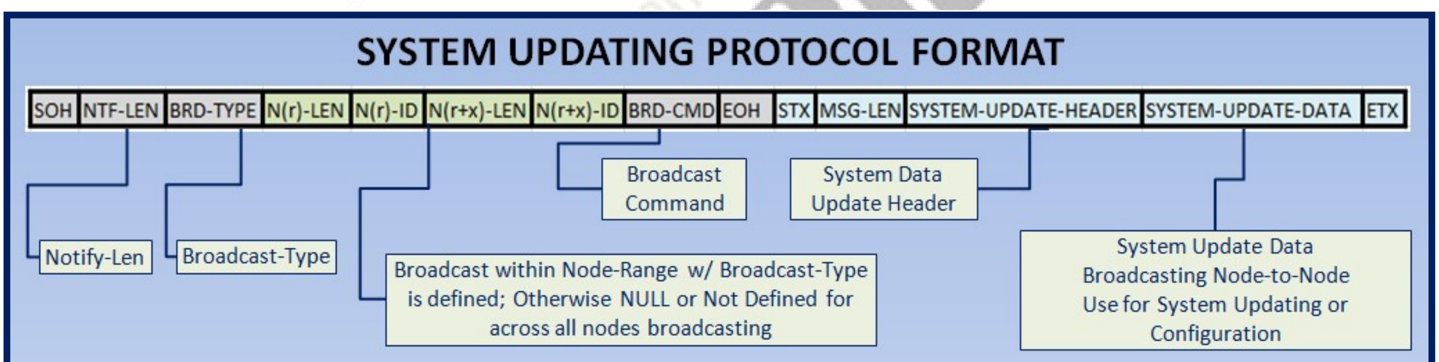


Figure-16: SYSTEM UPDATING PROTOCOL FORMAT

NODE'S PROPERTY & STATISTIC FIELDS FORMAT

N(x)-LEN	N(x)-ID	N(x)-SET	N(x)-PROP

N(x)-SET	N(x)-PROP	
Value	**Length**	**Value**
0x00	NULL	NULL
0x01	N(x)-PROP-LEN	N(x)-PROP-VAL
When N(x)-SET byte is 0x00; N(x)-PROP := NULL		

N(x)-LEN	N(x)-ID	N(x)-SET	N(x)-STAT

N(x)-SET	N(x)-STAT	
Value	**Length**	**Value**
0x00	NULL	NULL
0x01	N(x)-STAT-LEN	N(x)-STAT-VAL
When N(x)-SET byte is 0x00; N(x)-STAT := NULL		

Figure-17: NODE'S PROPERTY & STATISTIC FIELDS PROTOCOL FORMAT

Henry V. Pham

Conclusion

The **G-ROUTING ALGORITHM** is fast and robust searching algorithm to find the best routes for the **G-NETWORK** layouts. The **G-ROUTING ALGORITHM Methodology** is invented with State-of-the-Art and invented to work with any other networks from **simple networking nodes** layouts to the most **complicated machine networking nodes** layouts. This algorithm and the G-NETWORK layouts are ready for the future of the **3-D networks** or **multi-layer networks**, or **cell structure networks** like chemistry elements bonding networks.

The **G-ROUTING ALGORITHM Methodology** is invented to support any networks in some geometry in G-NETWORK layouts. The G-NETWORK layouts are controllable and manageable and self-grown when new nodes are added into the network or nodes are removed from the network. The G-NETWORK layouts must be defined in form of **Neighbor-to-Neighbors methodology** for the properties of the neighbor nodes of each network node. The Neighbor-to-Neighbors methodology requires smaller database compare to other existing network routing database. When a node is gone missing or unavailable for any reasons, adding a new node into the G-NETWORK layouts is much simple compare to the existing networks and only requires Neighbor-to-Neighbors notification of network property changes.

The **G-ROUTING ALGORITHM** is a promise for the future of networking and routing algorithm with the Neighbor-to-Neighbors networking technology. The **Cloud OS – Operating System**, **Cell ëMap Live Updates System**, and **Emergency Traffic Lights Routing System** inventions were intended to use the G-ROUTING ALGORITHM. The Neighbor-to-Neighbors networking technology provides the self-grown layout networks and can be more and more complexity and sophisticated in the manageable G-NETWORK form with multi-layers; only neighbor node knows its neighbor nodes. The **G-NETWORK** layout and **G-ROUTING ALGORITHM** provide the great secured and tied network than any other existing networks.