JET PROPULSION LABORATORY INTEROFFICE MEMORANDUM

SFB: 363-89-006
23 August 1989

TO: Time Warp Folks
FROM: Steven Bellenot & Phillip Hontalas

Pucks 2.0

Pucks underwent a transformation. By using the new extended virtual time, Pucks is able to correctly handle multiple collisions at the same simulation time and it no longer sends messages for now (i.e. send time equals receive time). This transformation also removed the last of the "twang" macro's (no more "^Ajunkjunkjunkjunkjunkjunkj ..." in the config files, no more "IVAR" or "MARG") and all other macro's. Also a number of logic preserving transformations was done to Pucks. The new Pucks has about a third less code and sequentially executes about a third faster than the old Pucks. Hence the average time per event has also dropped by a third. Pucks uses all the fields of the new extended virtual time. Because of the new virtual time the statistics are slightly different. (New Pucks: 414581 committed event messages, 370158 committed events; old Pucks: 412915 committed event messages, 366144 committed events.)

When Collisions Happen

The way in which the new Pucks handles multiple collisions at the same simulation time (a double in the new extended virtual time) is to have the collisions happen at different sequence times (sequence1 and sequence2 are unsigned longs in the new extended virtual time). Each kind of collision has its unique sequence2 time. Different collisions are at least 10 sequence2 units away so that the effects of one collision are known before the next collision is started.

Multiple collisions can happen at the same simulation time, but in order to do correct physics, Pucks serializes these collisions. For example, suppose puck A is going to hit both puck B and puck C at simulation time 10.0. Both B and C schedule a collision event with puck A at time 10.0, but they will have different sequence2 times for the collisions. Lets say puck B has time 10.0, 0, 1000 and puck C has time 10.0, 0, 1010. At time 10.0, 0, 1000 pucks A and B collide. By 10.0, 0, 1001, the sectors which A and B are in know about the new velocities which A and B have after the

collision. The sectors send this new information to everyone nearby at time 10.0, 0, 1002. Hence puck C receives this information well before his planned collision at time 10.0, 0, 1010 which he cancels. If with the new velocity, puck A (or puck B) will collide with C at 10.0, then C will schedule that collision at time 10.0, 1, 1010 (or with a different sequence2 time for puck B), increasing the sequence1 time.

The sequence2 time for a collision is unique to each pair of pucks. Sometimes (but not often) both pucks will schedule the same collision at the same time. In this case the sequence2 time for the two collisions are different. If puckWXYZ schedules a collision with puckABCD, then its sequence2 time will be WXYZ * 100000 + ABCD * 10 + 10000. On the other hand, if puckABCD schedules the collision it will happen at sequence2 time ABCD * 100000 + WXYZ * 10 + 10000. Note that the new Pucks requires pucks to be named "puckXXXX" where the X's must be decimal numbers, this is a new restriction for Pucks 2.0. Like old Pucks, cushions must be named "cushion_X_XX" and sectors must be named "sector_XX_XX". Collision times with cushion_X_YZ happen at sequence2 time XYZ * 10 + 10. Sector crossing times are either at sequence2 time 0 or the next sequence2 time.

Multiple Collision Example

Figure 1 shows the result of an eleven ball collision in the new Pucks. Figure 1A shows the starting position with the nine stationary pucks arranged in a touching square, and two touching pucks incoming at three o'clock, their arrows showing their initial velocity. Figure 1B shows the 11 pucks at time 0.178038 just before the collision. Figures C - N show the collision pairs as the new Pucks does them, all at simulation time 0.178038. Figures 0, P and Q show the net collision and how the pucks spread after the total collision.

Sequence1 = 0: At A all the pucks schedule collusions with either puck 9 or puck 10 or both. Pucks 6, 7 and 8 schedule this collusion at time 0.178038. Because of the ordering, puck 6 collides first with puck 9 (C). Next in line was puck 7's collision with puck 9 which gets cancelled. So puck 7 instead collides with puck 10 at 0.178038 (D). Thus both pucks 9 and 10 have new velocities so all other collisions are cancelled.

Sequence1 = 1: At D we start a new round of collisions with sequence1 incremented. The ordering of collisions has puck 3 colliding with puck 6 (E). Then puck 4 collides with puck 7 (F). Puck 9 scheduled a collision with puck 7, but it gets cancelled because 4 is lower than 9.

However, puck 8 does collide with puck 10 (G). We are still at simulation time 0.178038, and all the moving pucks but 9 have new velocities.

Sequence1 = 2. At G we again increment the sequence1 time to obtain the next collection of collisions. The ordering of collisions has puck 0 colliding with puck 3 (H). Followed by puck 1 colliding with puck 4 (I). Next puck 5 collides with puck 8 (J). And finally pucks 7 and 9 collide (J). We are still at simulation time 0.178038.

Sequence1 = 3. The last round of collisions are puck 2 with puck 5 (M) and puck 4 with puck 7 (N). Both N and O show the final velocities for all the pucks.

Since the start up is symmetric, one might expect the result to be also symmetric. However, as Pucks is currently done, at Figure 1B, pucks 6 and 8 (or even pucks 9 and 10) do not know that they are part of a multiple collision together. The other strange illusion is of very rigid pucks, for example that the collision between pucks 4 and 7 happen before the collision between pucks 7 and 9. However "real" pucks could never be packed so tight nor could their radii all be exactly the same. However, new Pucks does conserve momentum.

Other Transformations

The main loops of the sector, cushion and puck types were cut up into functions. The twang "TELL" macro, message selector field and standard_arguments stuff were replaced by the equivalent user interface stuff for TWOS 2.0. We expect that much of the increase in speed is due to not doing the additional clears and strcpys in the old TELL macro. Several instances of repeated code were replaced by single functions. The number of different message structures were reduced as were the size of the messages and the state. (The state size actually grew (see below) due to virtual time going from 4 to 16 bytes.) Some dead code was identified and deleted. A number of identifier names were shorten, for examples, "array_of_cancelled_indices" became "cannedMsgs" and the constant "EXAMINE_NEW_TRAJECTORY" became "NEW_TRAJECTORY". All the old style dot-h files for each object were deleted and the other include files were combined into one "pucktypes.h". (See file changes below.)

Pucks 2.0 uses a new config file format and there is a new "cfgen" (/usr/local/src/applications_2.1/pucks/cfgen2.1/cg) program for making these files. However the new Pucks can still read old Pucks config files.

By switching simulation time from an integer to a double, the new Pucks no longer scales virtual time. Thus simulations of much longer periods can be run. For example, the old Pucks benchmark ran to time 400,000,000 because it scaled by a million, while the new Pucks ran to time 400.0. Using floats for time and having multiple collisions at the same requires events which happened in the near past be consider as happening at now. (This is round-off error.) Currently an event must happen by at least TIME_FUDGE * now to be considered to be happening at now. TIME_FUDGE is 0.999999999. (And "eight 9's" isn't close enough to one for at least one config file.)

Performance Data

The new Pucks runs faster than the old pucks. Unfortunately the sequential run time dropped by a third while the Time Warp runs didn't improve by that much. The best speedup dropped by about a fifth from about 16 to about 13. This is consistent with much of the speed gain being from removing the old TELL macro's (the granularity went down). The benchmark runs were done with rcvq (maxnegacks) = 1, 10 and 30. The default was rcvq=10 with TW 2.1, but it had been rcvq=1 for the old Pucks TW 2.1 benchmark. However there wasn't much difference in the run times for these different rcvq's. (For large number of nodes rcvq=30 is faster. And rcvq=1 had troubles with both 6 and 8 nodes.) We have included the graphs for the rcvq=10 case, the stats files for all three cases and a comparison table for the different speedups.

Changing maxacks would likely have a larger effect on the run time. We didn't test this conjecture as maxacks works different in TW 2.1 than in either the published runs based on Test_2.1 nor as in TW 2.2 (the latter two work the same way). The memo's SFB: 363-89-005 and SFB:363-89-004 tested a version of Pucks different from both the old Pucks and the new Pucks, but it was closer to the old Pucks. However, the new Pucks seems to be "lazy" like the old Pucks (see graph below which was done using "test_2.1" not TW 2.1).

The striking difference between the old Pucks TW 2.1 benchmark and the new Pucks TW 2.1 benchmark is at the small number of nodes. The new Pucks is a third slower at 3 nodes and sends three times the reverse messages. The new Pucks sends hundreds of reverse messages up to 28 nodes while the old Pucks stopped sending any reverse messages at 12 nodes. This all may be due to the new VTime, a large number of fields in the state are VTime's in the new pucks and were doubles or integers in the old Pucks. A larger state makes memory more dear. (State sizes in

bytes on a Sun3: puck 3700 old, 4576 new; sector 5852 old, 6916 new; cushion 3272 old, 4136 new. Pucks could take advantage of TWOS's dynamic memory to greatly reduce the size of its states.)

File Changes:

```
Old Pucks
   Lines   Words   Bytes
       2       0       2 oldpucks/p_debug.h
       8      17     140 oldpucks/puck.h
       8      17     160 oldpucks/sector.h
       8      17     168 oldpucks/cushion.h
      13      39     347 oldpucks/puck.b
      19      35     396 oldpucks/formats.h
      37     204    1373 oldpucks/pucktab.c
      55     123     933 oldpucks/critsim.c
     147     511    4030 oldpucks/twc.h
     159     431    3113 oldpucks/pucktypes.h
     347     614    5724 oldpucks/p_debug_beh.c
     522    1303   11647 oldpucks/cushion_beh.c
    1633    4848   43218 oldpucks/sector_beh.c
    1813    5276   48423 oldpucks/puck_beh.c
    4771   13435  119674 total
New Pucks
   Lines   Words   Bytes
      35     204    1372 newpucks/pucktab.c
      53     123     936 newpucks/critsim.c
     134     365    2754 newpucks/pucktypes.h
     242     604    4612 newpucks/p_debug.c
     366    1117    8364 newpucks/cushion.c
    1196    3603   29117 newpucks/puck.c
    1238    3980   30217 newpucks/sector.c
    3264    9996   77372 total
```

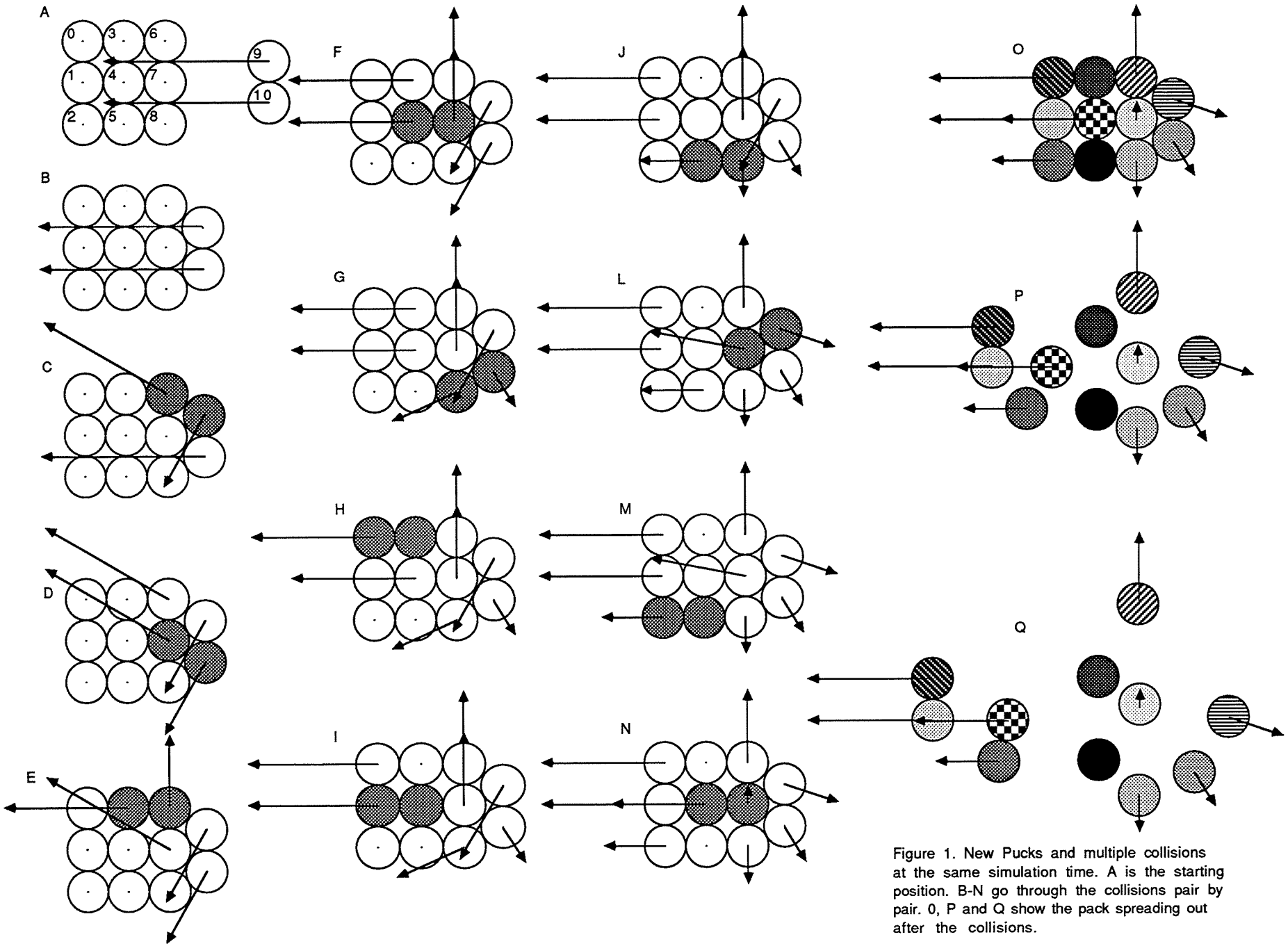
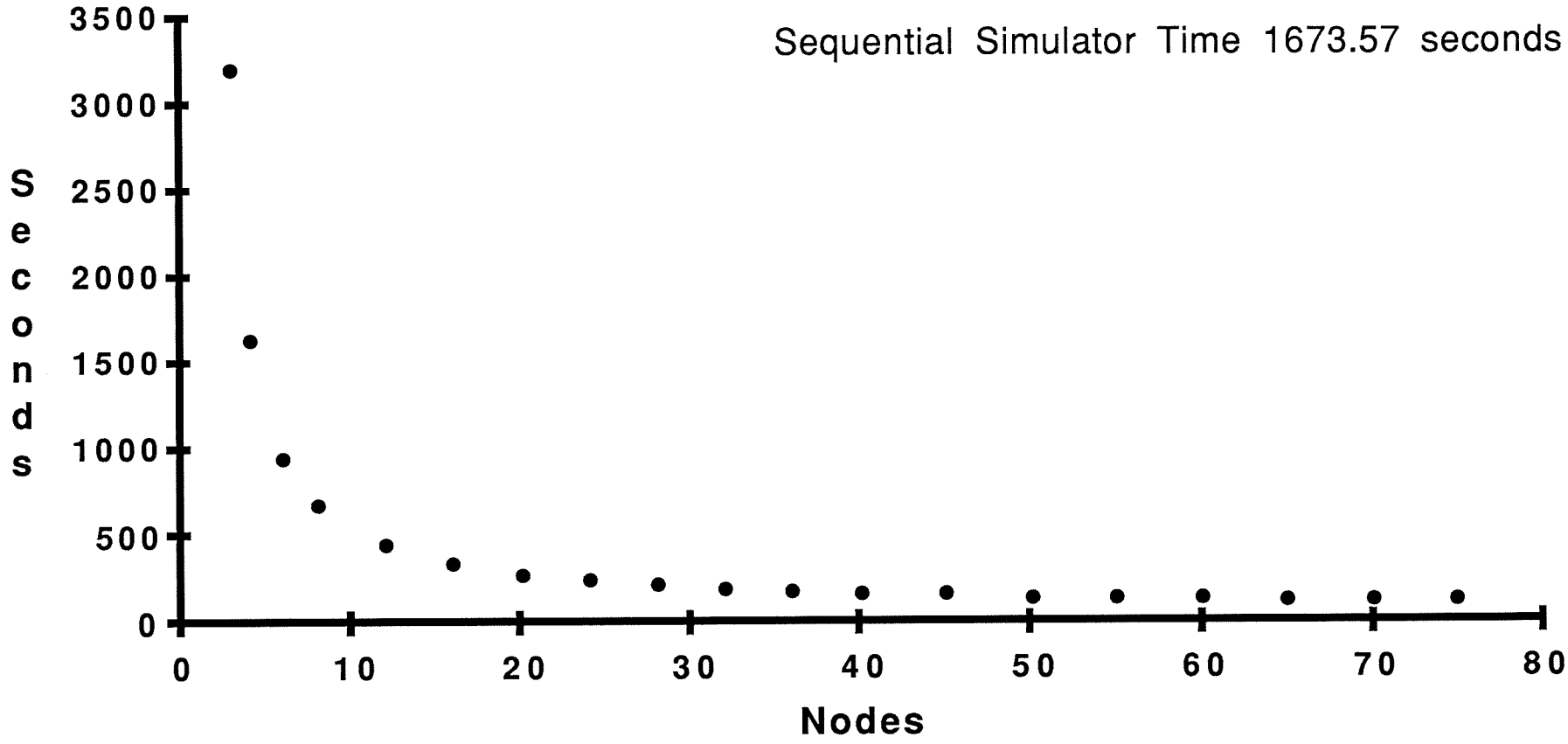


Figure 1. New Pucks and multiple collisions at the same simulation time. A is the starting position. B-N go through the collisions pair by pair. 0, P and Q show the pack spreading out after the collisions.

TW2.1 Pucks 2.0 Speedup as a function of rcvq			
Nodes	rcvq=1	rcvq=10	rcvq=30
3	0.55	0.52	0.54
4	1.01	1.02	1.04
6	1.76	1.75	1.76
8	2.42	2.43	2.43
12	3.68	3.71	3.69
16	4.84	4.86	4.86
20	5.83	5.85	5.85
24	6.71	6.76	6.74
28	7.41	7.49	7.44
32	8.15	8.16	8.16
36	9.02	9.02	9.02
40	9.58	9.54	9.58
45	10.07	10.08	10.07
50	10.92	11.10	11.16
55	11.59	11.73	11.73
60	12.15	12.16	12.14
65	12.41	12.66	12.66
70	12.17	12.32	12.84
75	12.31	12.70	12.94

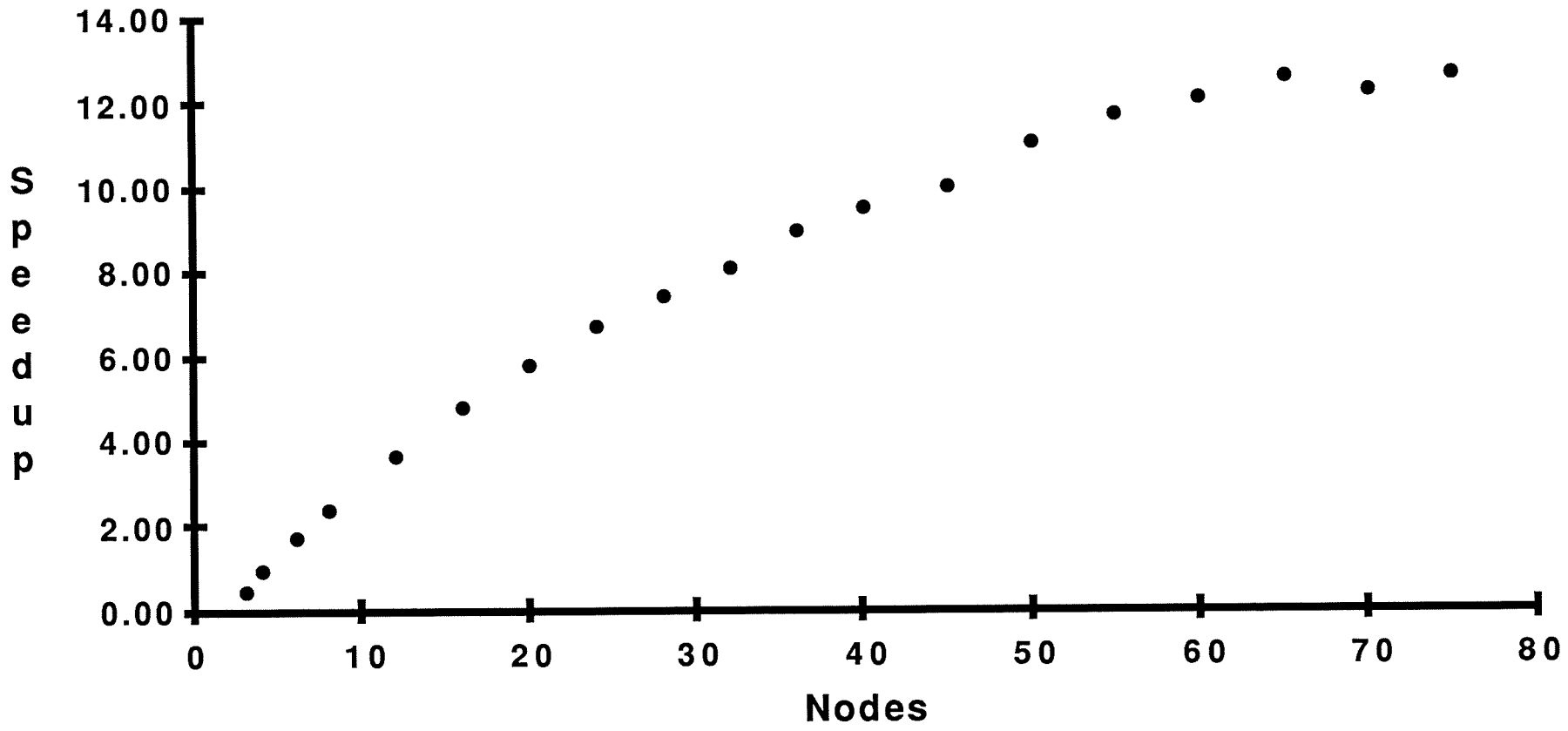
Pucks Timing Curve

Sequential Simulator Time 1673.57 seconds



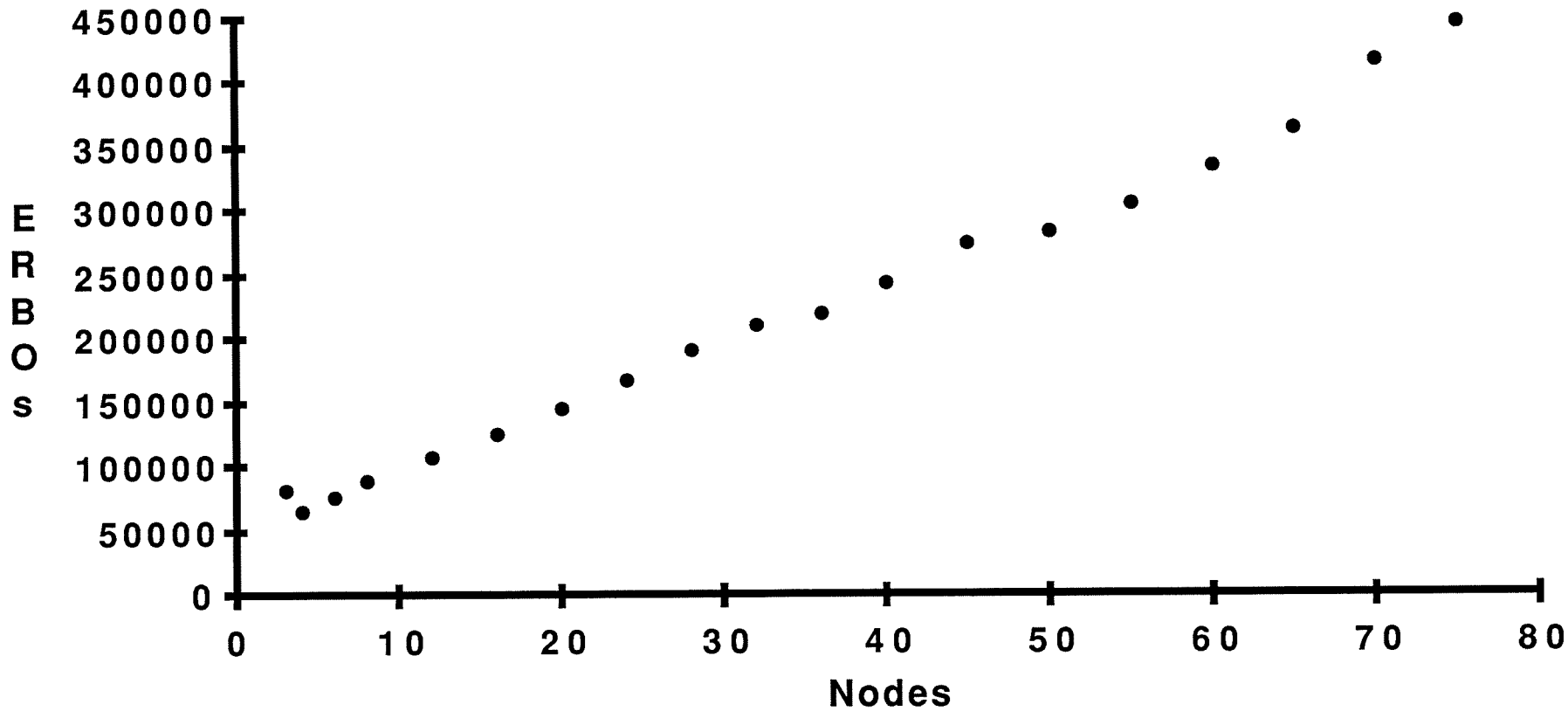
TW 2.1/Pucks 2.0 (rcvq=10)/4 runs per point/Butterfly (Chrysalis)/sfb 16 August 1989

Pucks Speedup Curve

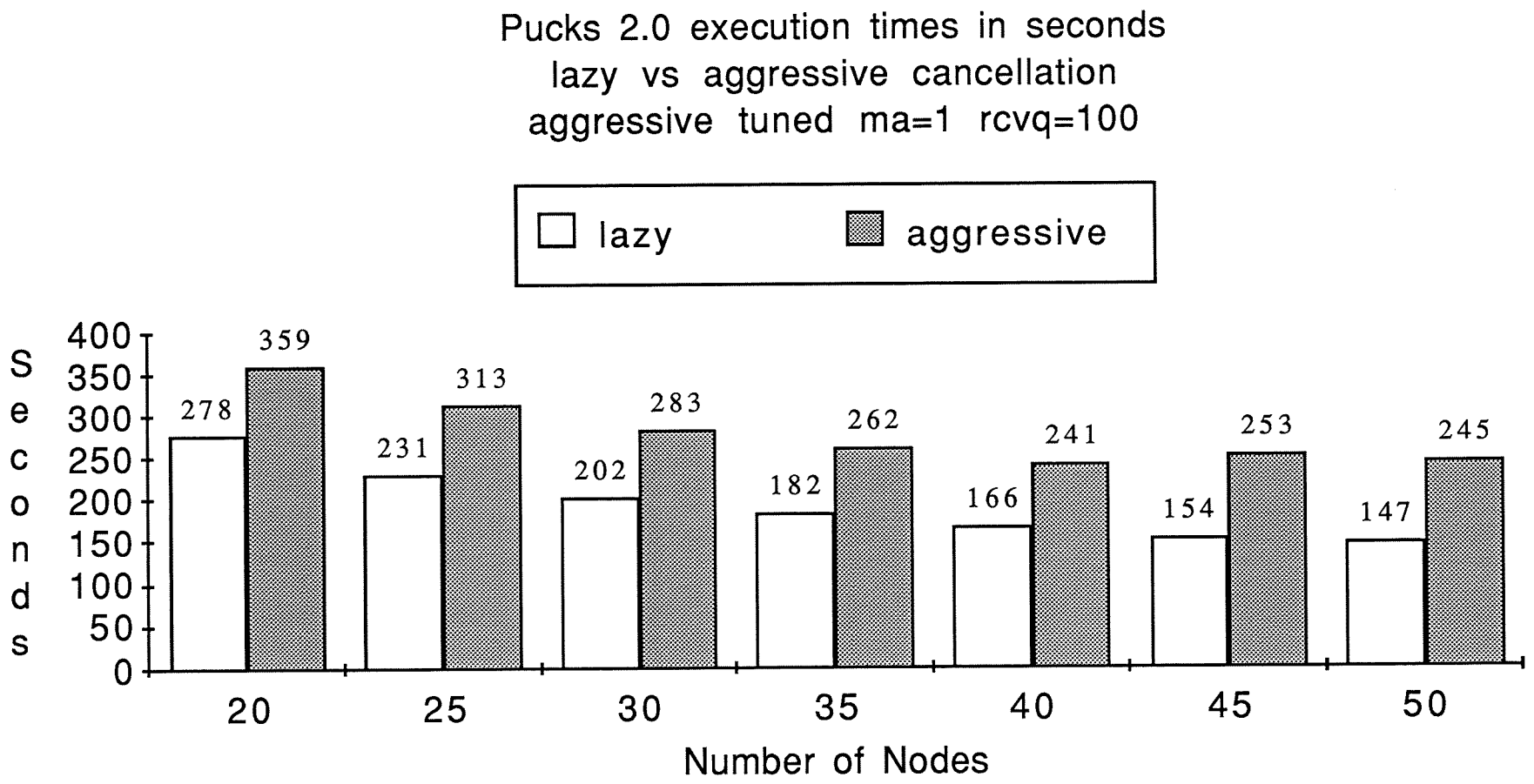


TW 2.1/Pucks 2.0 (rcvq=10)/4 runs per point/Butterfly (Chrysalis)/sfb 16 August 1989

Pucks ERBOs Curve



TW 2.1/Pucks 2.0 (rcvq=10)/4 runs per point/Butterfly (Chrysalis)/sfb 16 August 1989


Pucks 2.0 execution times in seconds
lazy vs aggressive cancellation
aggressive tuned ma=1 rcvq=100
lazy
aggressive
Seconds
400
350
300
250
200
150
100
50
0
278
359
231
313
202
283
182
262
166
241
154
253
147
245
20
25
30
35
40
45
50
Number of Nodes

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	Pucks 2.0 (rcvq=10) data for TW 2.1 Benchmark/4 runs per point/Butterfly (Chrysalis)/16 August 1989/gathered by Steven Bellenot																
2	nodes	ct:time	am:time	am:eposfs	am:enegfs	am:eposfr	am:enegfr	am:cemsqs	am:nestart	am:necomp	am:cevents	am:ecancel	am:nssave	am:nscom	am:nsequal	am:tells	am:eposrs
3	3	4	3203.09	534458	7750.3	534458	7750.3	414581	503102.5	453415.5	370158	7750.3	400453.3	370158	53266.3	937027.5	112126.8
4	4	4	1643.87	479387.3	10909.8	479387.3	10909.8	414581	465926.3	437483.8	370158	10909.8	408985.8	370158	28802	703915.8	53896.5
5	6	4	954.63	460057.8	15437.8	460057.8	15437.8	414581	467235.5	447387.5	370158	15437.8	424445.3	370158	23246.3	630632.5	30039
6	8	4	687.69	453282.8	20524.8	453282.8	20524.8	414581	477838.3	460699.8	370158	20524.8	439596.5	370158	21407.3	612775	18177
7	12	4	451.3	448469	27803.8	448469	27803.8	414581	497331.8	478945.3	370158	27803.8	460729.3	370158	18520	598086	6084.3
8	16	4	344.71	449300.5	32956	449300.5	32956	414581	518112.5	496217	370158	32956	477817.3	370158	18703.8	606698.8	1763.5
9	20	4	286.09	455207.3	39930.5	455207.3	39930.5	414581	542331	515964	370158	39930.5	495914	370158	20354	628499.8	695.8
10	24	4	247.45	460696	45964.3	460696	45964.3	414581	569170.8	538036.8	370158	45964.3	515859.8	370158	22481	653411.3	150.8
11	28	4	223.41	468922.3	54105	468922.3	54105	414581	597736	561800.5	370158	54105	537282	370158	24822.5	683277	236.3
12	32	4	205.1	474411.3	59809	474411.3	59809	414581	623206.8	582771.8	370158	59809	554966.8	370158	28109	705464.3	21.3
13	36	4	185.57	477956.3	63374.3	477956.3	63374.3	414581	635244	590596.8	370158	63374.3	563089	370158	27811.8	713787.5	1
14	40	4	175.37	486615.5	72033.5	486615.5	72033.5	414581	665310.3	615790	370158	72033.5	585634.8	370158	30459.3	745311	1
15	45	4	166.08	498542.8	83948	498542.8	83948	414581	703136.8	646716	370158	83948	613331.5	370158	33688.5	781212.3	13.8
16	50	4	150.8	501996.3	87415.3	501996.3	87415.3	414581	715584.5	655589	370158	87415.3	622376	370158	33517	795788	0
17	55	4	142.64	508498.5	93914.5	508498.5	93914.5	414581	743395.8	676596	370158	93914.5	640225.5	370158	36674.5	812007.3	3
18	60	4	137.67	518996.3	104415.3	518996.3	104415.3	414581	778959.5	705677	370158	104415.3	666010.3	370158	39970.8	844173	0
19	65	4	132.24	531573.3	116980	531573.3	116980	414581	814841.8	734249.8	370158	116980	692540	370158	42013.8	878535.3	12.3
20	70	4	135.85	557560.3	142841.5	557560.3	142841.5	414581	877785.3	788433.5	370158	142841.5	744819.3	370158	43918.3	953759.5	137.8
21	75	4	131.74	570295.5	155714.5	570295.5	155714.5	414581	911517.5	816728	370158	155714.5	771763	370158	45269	995564.5	0
22																	
23	Duplicate data for timing curve			Duplicate data for speed curve		Seq. Sim. Time		Calculated data for ERBO's curve									
24	Nodes	Seconds		Nodes	Speedup	1673.57		Nodes	ERBO's								
25	3	3203.09		3	0.52			3	83257.5								
26	4	1643.87		4	1.02			4	67325.8								
27	6	954.63		6	1.75			6	77229.5								
28	8	687.69		8	2.43			8	90541.8								
29	12	451.3		12	3.71			12	108787.3								
30	16	344.71		16	4.86			16	126059								
31	20	286.09		20	5.85			20	145806								
32	24	247.45		24	6.76			24	167878.8								
33	28	223.41		28	7.49			28	191642.5								
34	32	205.1		32	8.16			32	212613.8								
35	36	185.57		36	9.02			36	220438.8								
36	40	175.37		40	9.54			40	245632								
37	45	166.08		45	10.08			45	276558								
38	50	150.8		50	11.10			50	285431								
39	55	142.64		55	11.73			55	306438								
40	60	137.67		60	12.16			60	335519								
41	65	132.24		65	12.66			65	364091.8								
42	70	135.85		70	12.32			70	418275.5								
43	75	131.74		75	12.70			75	446570								

	R	S	T	U	V	W	X	Y	Z	AA	AB
1											
2	am:enegrs	am:eposrr	am:enegrr	am:numcr	am:numds	am:comcr	am:comds	am:cprobes	am:chits	am:cmisses	am:Objcpu
3	0	112126.8	0	0	0	0	0	1071961.5	1064376	7585.5	1588.33
4	0	53896.5	0	0	0	0	0	902280.5	898134.8	4145.8	1376
5	0	30039	0	0	0	0	0	857628.5	854615	3013.5	1328.5
6	0	18177	0	0	0	0	0	831865.5	828842.5	3023	1339.31
7	0	6084.3	0	0	0	0	0	827768.5	824217.8	3550.8	1358.69
8	0	1763.5	0	0	0	0	0	835000	830644.3	4355.8	1394.3
9	0	695.8	0	0	0	0	0	857314.5	852098.3	5216.3	1447.98
10	0	150.8	0	0	0	0	0	878775.5	872661.8	6113.8	1507.79
11	0	236.3	0	0	0	0	0	906287	899452.5	6834.5	1582.21
12	0	21.3	0	0	0	0	0	925613	917819	7794	1633.09
13	0	1	0	0	0	0	0	942291.5	933917.3	8374.3	1660.03
14	0	1	0	0	0	0	0	969110	960035.8	9074.3	1734.24
15	0	13.8	0	0	0	0	0	1010601.5	1000556.3	10045.3	1821.92
16	0	0	0	0	0	0	0	1023460	1013099	10361	1854.05
17	0	3	0	0	0	0	0	1043802.5	1032420.8	11381.8	1910.22
18	0	0	0	0	0	0	0	1079273.5	1066811.8	12461.8	1997.28
19	0	12.3	0	0	0	0	0	1122974	1109670.8	13303.3	2090.52
20	0	137.8	0	0	0	0	0	1207706	1194422.8	13283.3	2278.88
21	0	0	0	0	0	0	0	1246518.5	1233148.3	13370.3	2377.93
22											
23											
24											
25											
26											
27											
28											
29											
30											
31											
32											
33											
34											
35											
36											
37											
38											
39											
40											
41											
42											
43											

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	Pucks 2.0 (rcvq=30) data for TW 2.1 Benchmark/4 runs per point/Butterfly (Chrysalis)/16 August 1989/gathered by Steven Bellenot																
2	nodes	ct:time	am:time	am:eposfs	am:enegfs	am:eposfr	am:enegfr	am:cemsgs	am:nestart	am:necomp	am:cevents	am:ecancel	am:nssave	am:nscom	am:nsequal	am:tells	am:eposrs
3	3	4	3084.66	515194	7351.5	515194	7351.5	414581	487810	443547.3	370158	7351.5	396932	370158	46919.3	875821.5	93261.5
4	4	4	1613.63	473533.5	10678.5	473533.5	10678.5	414581	460947.8	431537.8	370158	10678.5	407062	370158	24779.8	687143.3	48274
5	6	4	949.1	458273	15627	458273	15627	414581	466364.3	445956.3	370158	15627	424220.8	370158	22039.5	627649	28065
6	8	4	687.81	452974.8	20464	452974.8	20464	414581	477813	459851.8	370158	20464	439318.8	370158	20837	613279.8	17929.8
7	12	4	453.49	449125.5	27924.3	449125.5	27924.3	414581	498046.3	479261.8	370158	27924.3	461016.8	370158	18549	601580.5	6620.3
8	16	4	344.63	449187.8	33093.8	449187.8	33093.8	414581	518012.3	496190.8	370158	33093.8	477898.5	370158	18596.3	605407	1513
9	20	4	286.04	455059	39801	455059	39801	414581	541891.3	515583.3	370158	39801	495590.3	370158	20297	627804.8	677
10	24	4	248.15	460259	45576.8	460259	45576.8	414581	568591	537611.5	370158	45576.8	515546.5	370158	22369	652061.8	101.3
11	28	4	225.06	469178.3	54281.5	469178.3	54281.5	414581	597944.5	562078.5	370158	54281.5	537593.8	370158	24788.8	684388.8	315.8
12	32	4	205.04	474449	59860	474449	59860	414581	623375.5	582770.5	370158	59860	554997.3	370158	28077.3	705519.3	8
13	36	4	185.51	478484.8	63903	478484.8	63903	414581	635685.3	590996.5	370158	63903	563334.8	370158	27965.8	714551	0.8
14	40	4	174.69	486133.3	71552.3	486133.3	71552.3	414581	664021	614534	370158	71552.3	584419	370158	30419	744078.3	0
15	45	4	166.25	498655.5	84029.3	498655.5	84029.3	414581	703277.5	646699	370158	84029.3	613287.8	370158	33715.3	781850.3	45.3
16	50	4	149.98	501221.3	86640.3	501221.3	86640.3	414581	714047.3	654390.8	370158	86640.3	621204.5	370158	33490.3	793607.5	0
17	55	4	142.62	508484.5	93903.5	508484.5	93903.5	414581	742591.8	676140.8	370158	93903.5	639772.8	370158	36672	811386.3	0
18	60	4	137.8	519443.5	104862.5	519443.5	104862.5	414581	779465	706064.3	370158	104862.5	666414	370158	39954.3	845034	0
19	65	4	132.16	529948.5	115306.8	529948.5	115306.8	414581	810158.8	730019.8	370158	115306.8	688796.5	370158	41527.3	873568.5	60.8
20	70	4	130.33	546402.8	131774.8	546402.8	131774.8	414581	844844.8	759459.8	370158	131774.8	718436.3	370158	41327.5	919745.3	47
21	75	4	129.36	564192.3	149608.3	564192.3	149608.3	414581	892945.3	799943	370158	149608.3	756249	370158	43998	975064	3
22																	

	Duplicate data for timing curve			Duplicate data for speed curve		Seq. Sim. Time		Calculated data for ERBO's curve	
24	Nodes	Seconds		Nodes	Speeedup	1673.57		Nodes	ERBO's
25	3	3084.66		3	0.54			3	73389.3
26	4	1613.63		4	1.04			4	61379.8
27	6	949.1		6	1.76			6	75798.3
28	8	687.81		8	2.43			8	89693.8
29	12	453.49		12	3.69			12	109103.8
30	16	344.63		16	4.86			16	126032.8
31	20	286.04		20	5.85			20	145425.3
32	24	248.15		24	6.74			24	167453.5
33	28	225.06		28	7.44			28	191920.5
34	32	205.04		32	8.16			32	212612.5
35	36	185.51		36	9.02			36	220838.5
36	40	174.69		40	9.58			40	244376
37	45	166.25		45	10.07			45	276541
38	50	149.98		50	11.16			50	284232.8
39	55	142.62		55	11.73			55	305982.8
40	60	137.8		60	12.14			60	335906.3
41	65	132.16		65	12.66			65	359861.8
42	70	130.33		70	12.84			70	389301.8
43	75	129.36		75	12.94			75	429785

	R	S	T	U	V	W	X	Y	Z	AA	AB
1											
2	am:enegrs	am:eposrr	am:enegrr	am:numcr	am:numds	am:comcr	am:comds	am:cprobes	am:chits	am:cmisses	am:Objcpu
3	0	93261.5	0	0	0	0	0	1006389.5	998887	7502.5	1526.18
4	0	48274	0	0	0	0	0	882433.5	878369.3	4064.3	1358.2
5	0	28065	0	0	0	0	0	851158	848207.3	2950.8	1324.58
6	0	17929.8	0	0	0	0	0	830999	828004	2995	1338.48
7	0	6620.3	0	0	0	0	0	830216.5	826658	3558.5	1362.98
8	0	1513	0	0	0	0	0	834445	830086.5	4358.5	1394.43
9	0	677	0	0	0	0	0	856797.5	851582.5	5215	1448.79
10	0	101.3	0	0	0	0	0	877210.5	871099	6111.5	1508.3
11	0	315.8	0	0	0	0	0	907180.5	900347	6833.5	1585.34
12	0	8	0	0	0	0	0	925728.5	917934	7794.5	1635.64
13	0	0.8	0	0	0	0	0	944095.5	935719.8	8375.8	1661.47
14	0	0	0	0	0	0	0	967347.5	958272.3	9075.3	1733.34
15	0	45.3	0	0	0	0	0	1011107.5	1001057.5	10050	1825.62
16	0	0	0	0	0	0	0	1020739	1010379.3	10359.8	1854.43
17	0	0	0	0	0	0	0	1043641	1032264.8	11376.3	1912.66
18	0	0	0	0	0	0	0	1080768	1068307.8	12460.3	2002.56
19	0	60.8	0	0	0	0	0	1117611	1104315.5	13295.5	2081.37
20	0	47	0	0	0	0	0	1170781	1157519.8	13261.3	2197.58
21	0	3	0	0	0	0	0	1226389	1213030.5	13358.5	2332.18
22											
23											
24											
25											
26											
27											
28											
29											
30											
31											
32											
33											
34											
35											
36											
37											
38											
39											
40											
41											
42											
43											

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	Pucks 2.0 (rcvq=1) data for TW 2.1 Benchmark/4 runs per point/Butterfly (Chrysalis)/22 August 1989/gathered by Steven Bellenot																
2	nodes	ct:time	am:time	am:eposfs	am:eneqfs	am:eposfr	am:eneqfr	am:cemsgs	am:nestart	am:necomp	am:cevents	am:ecancel	am:nssave	am:nscom	am:nsequal	am:tells	am:eposrs
3	3	4	3055.32	532297.3	7621.8	532297.3	7621.8	414581	496879	446587	370158	7621.8	400527.8	370158	46363.3	865754	110094.5
4	4	4	1659.74	483215.5	10667.5	483215.5	10667.5	414581	467369.8	438108.8	370158	10667.5	409178.3	370158	29234.5	688804.8	57967
5	6 **fails	4	952.99	461317.5	15406.3	461317.5	15406.3	414581	468270.8	448100.8	370158	15406.3	424897.3	370158	23507.5	623450.5	31330.3
6	8 **fails	4	692.49	455446.5	20552	455446.5	20552	414579.5	480718.5	462213.5	370156.8	20552	440328.5	370158	22189	615526.3	20313.5
7	12	4	454.17	450175.8	28202.3	450175.8	28202.3	414581	501058.3	481483	370158	28202.3	462427	370158	19360	604564.3	7392.5
8	16	4	345.57	450113.3	33106.8	450113.3	33106.8	414581	520825.8	497765.8	370158	33106.8	478720.3	370158	19349.5	609860.5	2425.5
9	20	4	287	455807.3	40052	455807.3	40052	414581	545227.8	517456	370158	40052	496959.5	370158	20800.5	630968	1174.3
10	24	4	249.56	461407	46500.3	461407	46500.3	414581	573579.8	540721	370158	46500.3	517959.8	370158	23065.3	656611.5	325.8
11	28	4	225.93	469825	54892.3	469825	54892.3	414581	602715.3	564852.8	370158	54892.3	539791.3	370158	25365.5	686725.3	351.8
12	32	4	205.3	474969.5	60367.5	474969.5	60367.5	414581	628246.8	585345.8	370158	60367.5	556930.8	370158	28719	707649	21
13	36	4	185.62	479173.3	64562.5	479173.3	64562.5	414581	642230.8	594806.5	370158	64562.5	566470.8	370158	28639.8	718445.8	29.8
14	40	4	174.73	486680.5	72098.5	486680.5	72098.5	414581	669242.5	616977.8	370158	72098.5	586116.3	370158	31165.5	744984.3	1
15	45	4	166.12	500253.3	85643.5	500253.3	85643.5	414581	712337	652555	370158	85643.5	617805.5	370158	35053.5	787252.3	28.8
16	50	4	153.2	502787.8	88206.8	502787.8	88206.8	414581	723661.8	660041.3	370158	88206.8	625545.8	370158	34799.5	798511	0
17	55	4	144.42	510593.8	96004	510593.8	96004	414581	753522	682615	370158	96004	644750.3	370158	38168.8	818499.3	8.8
18	60	4	137.75	521205.5	106624.5	521205.5	106624.5	414581	791022	713031.5	370158	106624.5	671503.3	370158	41832.3	851532	0
19	65	4	134.83	534786	120205	534786	120205	414581	830145.8	743995.3	370158	120205	700269.5	370158	44029.8	888051.8	0
20	70	4	137.5	561996.8	147415.8	561996.8	147415.8	414581	893703	798904.3	370158	147415.8	753255	370158	45953.3	967523.8	0
21	75	4	135.95	578697.5	164097	578697.5	164097	414581	942017.5	839869.3	370158	164097	791423.3	370158	48750	1022442.8	19.5
22																	
23	Duplicate data for timing curve			Duplicate data for speed curve		Seq. Sim. Time		Calculated data for ERBO's curve									
24	Nodes	Seconds		Nodes	Speedup	1673.57		Nodes	ERBO's								
25	3	3055.32		3	0.55			3	76429								
26	4	1659.74		4	1.01			4	67950.8								
27	6	952.99		6	1.76			6	77942.8								
28	8	692.49		8	2.42			8	92056.7								
29	12	454.17		12	3.68			12	111325								
30	16	345.57		16	4.84			16	127607.8								
31	20	287		20	5.83			20	147298								
32	24	249.56		24	6.71			24	170563								
33	28	225.93		28	7.41			28	194694.8								
34	32	205.3		32	8.15			32	215187.8								
35	36	185.62		36	9.02			36	224648.5								
36	40	174.73		40	9.58			40	246819.8								
37	45	166.12		45	10.07			45	282397								
38	50	153.2		50	10.92			50	289883.3								
39	55	144.42		55	11.59			55	312457								
40	60	137.75		60	12.15			60	342873.5								
41	65	134.83		65	12.41			65	373837.3								
42	70	137.5		70	12.17			70	428746.3								
43	75	135.95		75	12.31			75	469711.3								

	R	S	T	U	V	W	X	Y	Z	AA	AB
1											
2	am:enegrs	am:eposrr	am:enegrr	am:numcr	am:numds	am:comcr	am:comds	am:cprobes	am:chits	am:cmisses	am:Objcpu
3	0	110094.5	0	0	0	0	0	1060744.5	1052924.8	7819.8	1518.95
4	0	57967	0	0	0	0	0	916134	911743.3	4390.8	1365.06
5	0	31330.3	0	0	0	0	0	862250.5	859047.3	3203.3	1323.25
6	0	20313.5	0	0	0	0	0	840235.5	837079	3156.5	1342.15
7	0	7392.5	0	0	0	0	0	834065	830465.8	3599.3	1366.32
8	0	2425.5	0	0	0	0	0	838091	833710.8	4380.3	1398.18
9	0	1174.3	0	0	0	0	0	859517	854287.5	5229.5	1451.7
10	0	325.8	0	0	0	0	0	881134.5	875018.5	6116	1514.5
11	0	351.8	0	0	0	0	0	909334	902497.5	6836.5	1588.95
12	0	21	0	0	0	0	0	927397	919600.5	7796.5	1639.12
13	0	29.8	0	0	0	0	0	946311	937936	8375	1668.42
14	0	1	0	0	0	0	0	969153.5	960079.8	9073.8	1734.58
15	0	28.8	0	0	0	0	0	1016168.5	1006110.3	10058.3	1837.28
16	0	0	0	0	0	0	0	1025979.5	1015609.8	10369.8	1866.06
17	0	8.8	0	0	0	0	0	1050432	1039043.5	11388.5	1927.83
18	0	0	0	0	0	0	0	1086431	1073955.8	12475.3	2017.95
19	0	0	0	0	0	0	0	1133372	1120062.3	13309.8	2118.7
20	0	0	0	0	0	0	0	1221859.5	1208566.5	13293	2313.86
21	0	19.5	0	0	0	0	0	1274157.5	1260783.8	13373.8	2446.61
22											
23											
24											
25											
26											
27											
28											
29											
30											
31											
32											
33											
34											
35											
36											
37											
38											
39											
40											
41											
42											
43											