11.2.1 against 11.1.1

value-only workload

Michael Karg, Cardano Performance & Tracing
2026-11-12

Contents

Manifest 2
Analysis 3
- Resource Usage 3
- Anomaly control 3
- Forging 4
- Individual peer propagation 4
- End-to-end propagation 4
- Ledger Snapshots 5
- Transaction Processing 5

Appendix A: charts 6
- Cluster performance charts 6

Appendix B: data dictionary 21
- Block propagation metrics 21
- Cluster performance metrics 22

Manifest

We compare `11.1.1` (Conway) and `11.2.1` (Conway) relative to `11.1.1` (Conway), under value-only workload.
NB: LO stands for log object, i.e. a trace message emitted by a node.

	11.1.1	11.2.1
Analysis date	2026-11-10	2026-11-11
Cluster system start date	2026-09-05	2026-11-10
Cluster system start time	21:53:06	17:03:06
Identifier	11.1.1	11.2.1
Run batch	11.1.1	11.2.1
GHC version	9.6.7	9.6.7
cardano-node	11.1.1 @ 07d5a1c	11.2.1 @ eee0e14
ouroboros-consensus	4.2.1.0 @ 82ecba3	5.4.0.0 @ 808e400
ouroboros-network	1.2.0.0 @ c45735a	2.0.0.0 @ 4646039
cardano-ledger-core	1.21.0.0 @ f649f97	1.23.0.0 @ 95b3690
plutus-core	1.65.0.0 @ b2db512	1.71.0.0 @ 261766a
cardano-crypto	1.3.0 @ a741501	1.4.0 @ ac2e12a
cardano-prelude	0.2.2.0 @ 2b6092c	0.2.2.0 @ 2b6092c
Era	conway	conway
Delegation map size	1000000	1000000
Stuffed UTxO size	4000000	4000000
Final UTxO size	6536069	6536069
DRep count	10000	10000
Extra tx payload	100	100
Tx inputs	2	2
Tx Outputs	2	2
TPS	12	12
Transaction count	768000	768000
Plutus script	---	---
Machines	52	52
Filters applied, count	3	3
LO emitted / host	4111880.885	4075414.846
LO analysed / host	1807215.519	1795168.481
Host run time, s	63943.730	63940.703
Host log line rate, Hz	64.305	63.738
LO analysed, total	93975207	93348761
Run time, s	63958	63950
Analysed run duration, s	48033	48027
Run time efficiency	0.751	0.751
Node start spread, s	21.883	16.814
Node stop spread, s	4.243	2.705
Slots analysed	48030	48025
Blocks analysed	2206	2243
Blocks rejected	898	837

Analysis

Resource Usage

	11.1.1	11.2.1	Δ	Δ%
Forge loop starts, units	0.99993	0.99994	0.0	0
Process CPU usage, %	2.6748	2.5419	−0.1329	−5
RTS GC CPU usage, %	0.3075	0.3063	−0.0012	−0.4
RTS Mutator CPU usage, %	2.3683	2.2365	−0.1318	−6
Major GCs, events	0.0008	0.0009	0.0001	14
Minor GCs, events	0.689	0.6531	−0.0359	−5
Kernel RSS, MiB	8256.3	7064.0	−1192.3	−14
RTS heap size, MiB	8186.2	6988.0	−1198.2	−15
RTS live GC dataset, MiB	4208.3	3170.6	−1037.6	−25
RTS alloc rate, MiB/s	21.315	20.023	−1.2918	−6
Filesystem reads, KiB/s	0.0257	6.1e-5	−0.0256	−100
Filesystem writes, KiB/s	238.87	238.97	0.0944	0.0
CPU 85% peaks, ep. trans., slots	1.846	2.0386	0.1926	10
CPU 85% peaks, other, slots	1.2443	1.1496	−0.0946	−8

Anomaly control

	11.1.1	11.2.1	Δ	Δ%
Blocks per host, blocks	61.923	60.904	−1.0192	−2
Filtered to chained blocks, :	0.7096	0.7295	0.0199	3
Chained to forged blocks, :	0.9636	0.9712	0.0076	0.8
Slot battles, blocks	0.02949	0.02028	−0.0092	−31
Height battles, blocks	0.01068	0.00734	−0.00335	−31
Block size, Bytes	88956	88965	8.6358	0.0

Forging

	11.1.1	11.2.1	Δ	Δ%
Started forge loop iteration, s	0.00091	0.001	0.0	0
Acquired block context, s	6.2e-5	5.9e-5	−0.0	0
Acquired ledger state, s	0.00011	0.00011	−0.0	0
Acquired ledger view, s	3.5e-5	3.4e-5	−0.0	0
Leadership check duration, s	0.00042	0.00041	−0.0	0
Ledger ticking, s	0.01826	0.01894	0.00068	4
Mempool snapshotting, s	0.04206	0.03992	−0.00214	−5
Leadership to forged, s	0.00081	0.00129	0.00048	59
Forged to announced, s	0.00066	0.00062	−0.0	0
Forged to sending, s	0.00571	0.00526	−0.00045	−8
Forged to self-adopted, s	0.06647	0.06572	−0.00076	−1
Slot start to announced, s	0.06333	0.06239	−0.00094	−1

Individual peer propagation

	11.1.1	11.2.1	Δ	Δ%
First peer notice, s	0.0651	0.06392	−0.00118	−2
First peer fetch, s	0.07587	0.07387	−0.002	−3
Notice to fetch request, s	0.00144	0.00138	−0.0	0
Fetch duration, s	0.38649	0.37071	−0.01578	−4
Fetched to announced, s	0.00087	0.00086	−0.0	0
Fetched to sending, s	0.04168	0.04255	0.00087	2
Fetched to adopted, s	0.06857	0.06902	0.00045	0.7

End-to-end propagation

	11.1.1	11.2.1	Δ	Δ%
0.50 adoption, s	0.60771	0.6225	0.01479	2
0.80 adoption, s	1.05466	0.97949	−0.07516	−7
0.90 adoption, s	1.07285	0.99645	−0.0764	−7
0.92 adoption, s	1.07718	1.00105	−0.07613	−7
0.94 adoption, s	1.08171	1.00559	−0.07611	−7
0.96 adoption, s	1.08706	1.01032	−0.07674	−7
0.98 adoption, s	1.09376	1.01859	−0.07517	−7
1.00 adoption, s	1.10904	1.03671	−0.07232	−7

Ledger Snapshots

	11.1.1	11.2.1	Δ	Δ%
Taken, count	11.0	11.0	0.0	0
Duration, s	18.763	18.688	−0.075	−0.4
Live after majGC, during, MiB	2601.8	2156.0	−445.77	−17
CPU 85% peaks, snapshot, slots	18.409	18.008	−0.4011	−2
Live after majGC, outside, MiB	2122.7	2128.4	5.6898	0.3

Transaction Processing

	11.1.1	11.2.1	Δ	Δ%
Txns processed, count	409483	399851	−9632.3	−2
Txns accepted, count	311063	302149	−8913.3	−3
Txns rejected, count	98421	97702	−719.04	−0.7
Reject ratio, :	0.2233	0.222	−0.0013	−0.6
Mempool fill level, count	461.34	466.04	4.7033	1
Proc CPU ÷ txns, ms	3.3091	3.2419	−0.0672	−2
GC CPU ÷ txns, ms	0.3918	0.4049	0.0131	3
Mut CPU ÷ txns, ms	2.9185	2.8381	−0.0804	−3

Appendix A: charts

Cluster performance charts

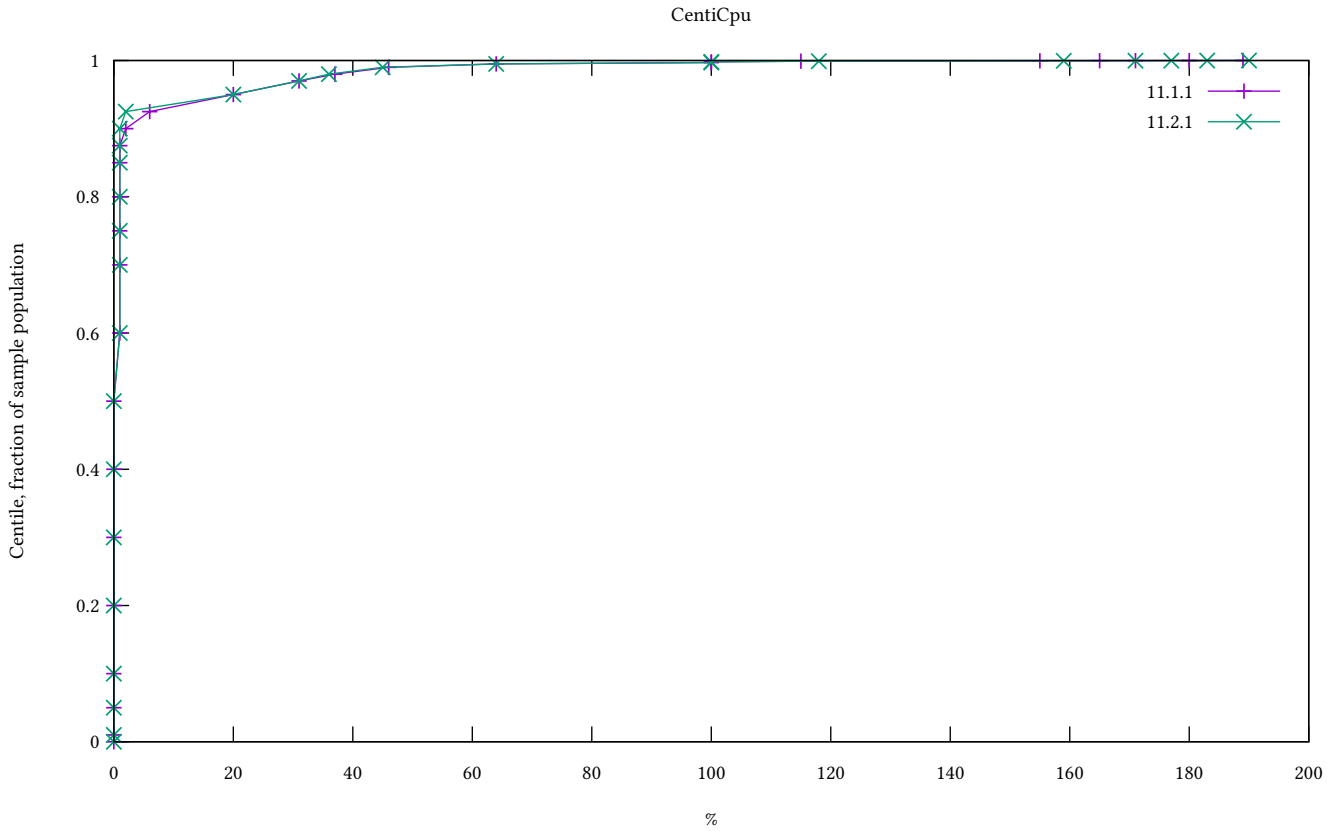


Figure 1: Process CPU usage

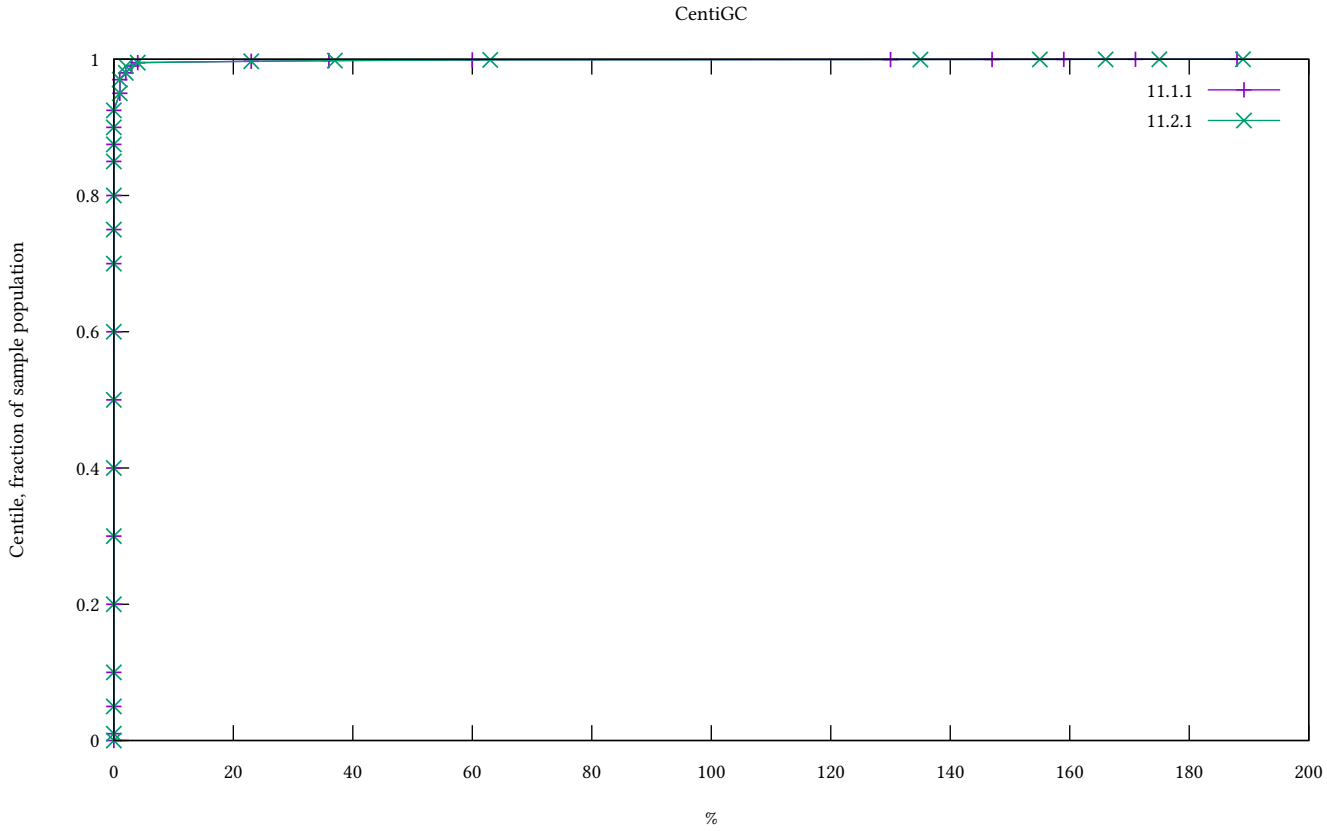


Figure 2: RTS GC CPU usage

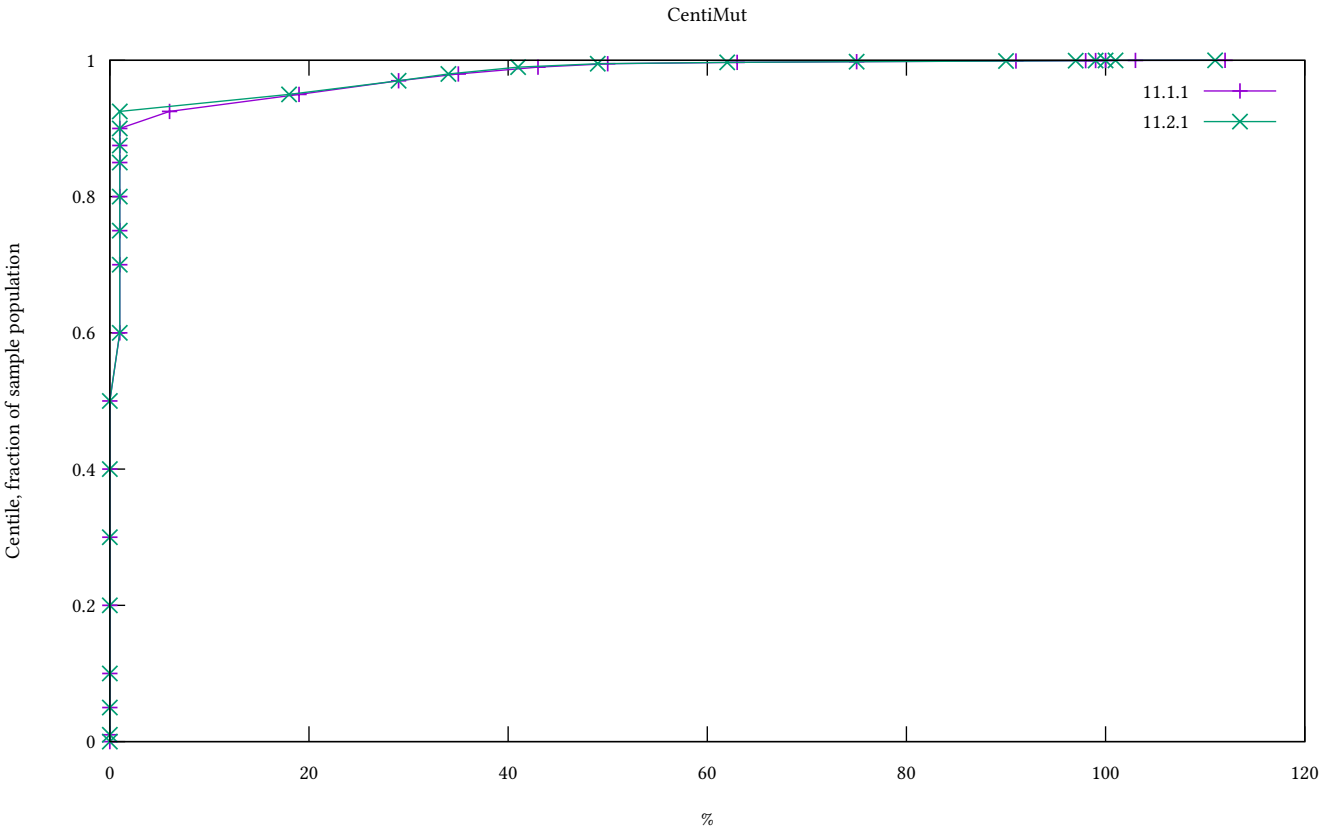


Figure 3: RTS Mutator CPU usage

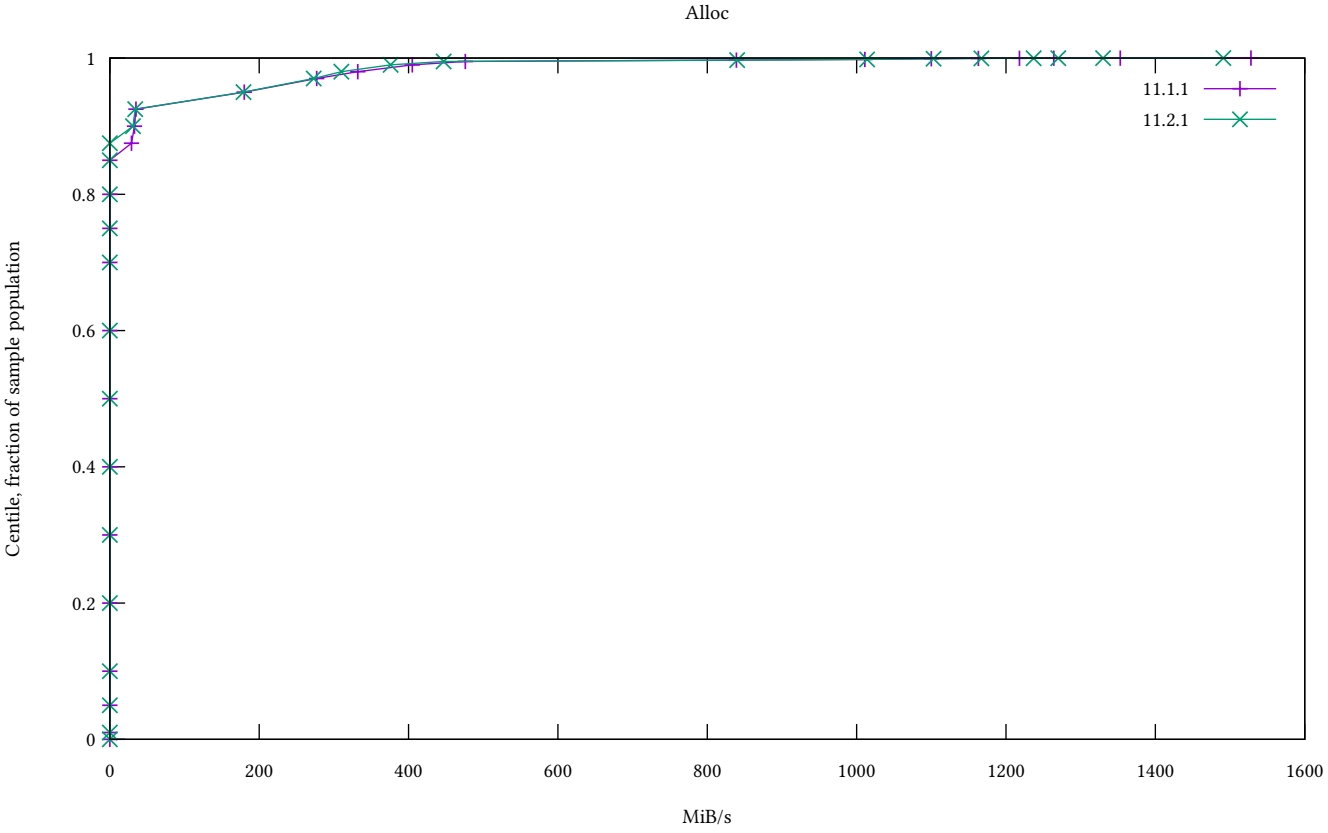


Figure 4: RTS alloc rate

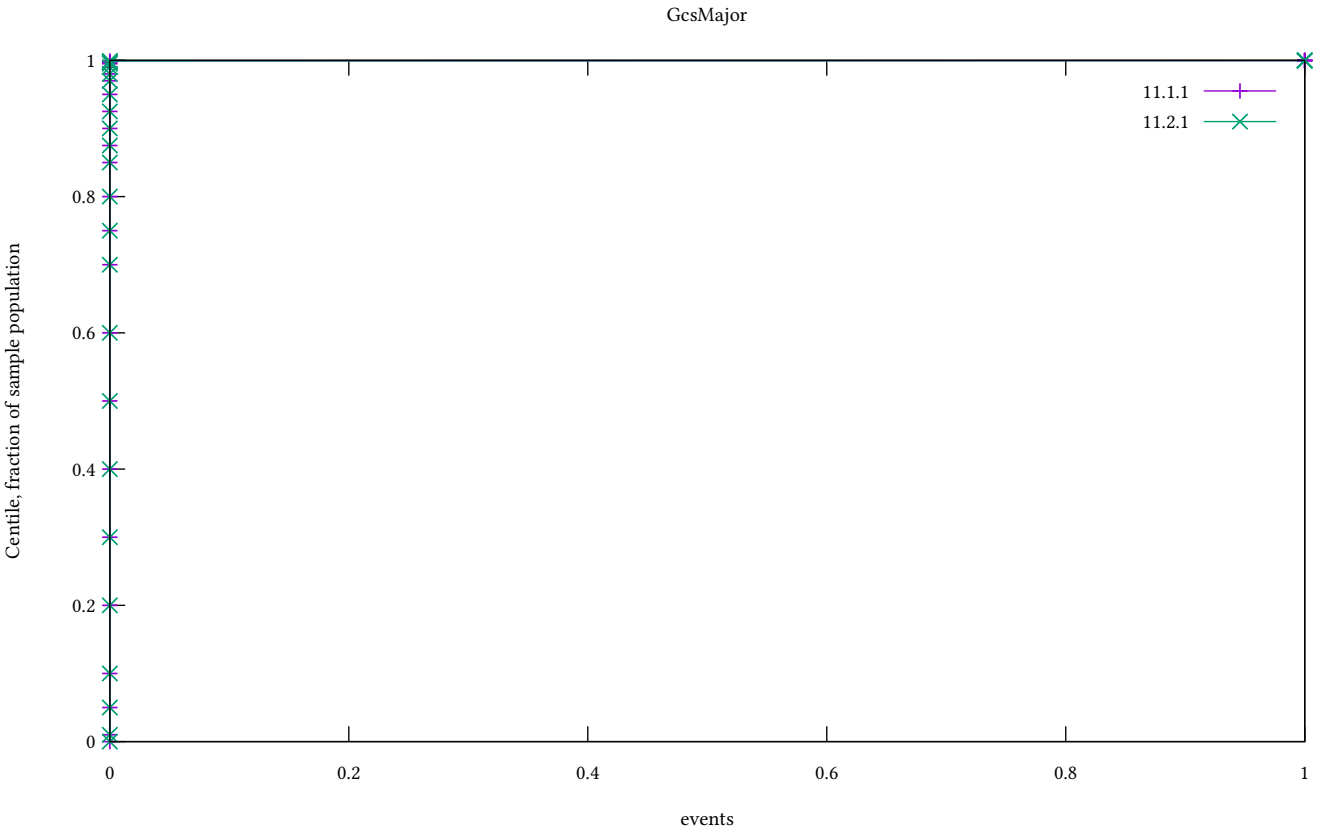


Figure 5: Major GCs

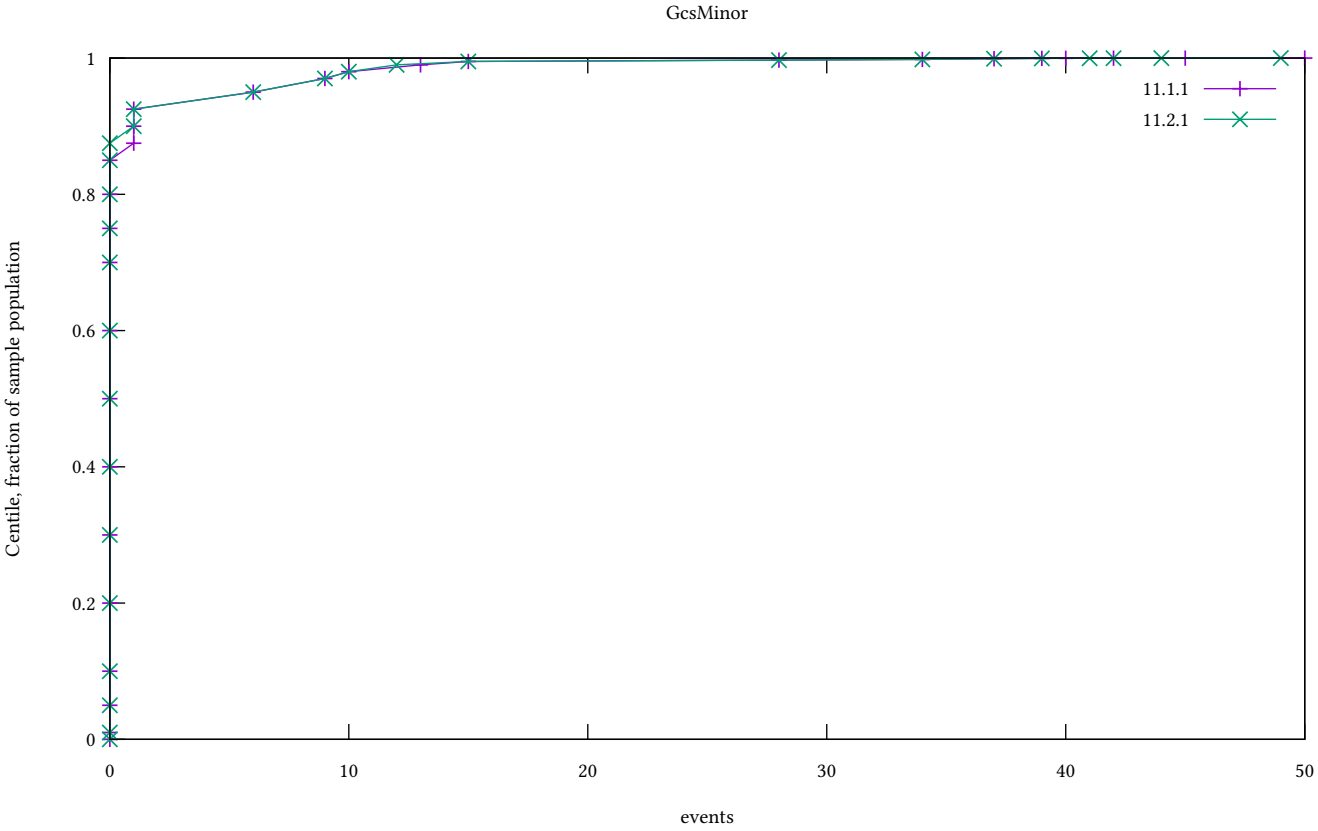


Figure 6: Minor GCs

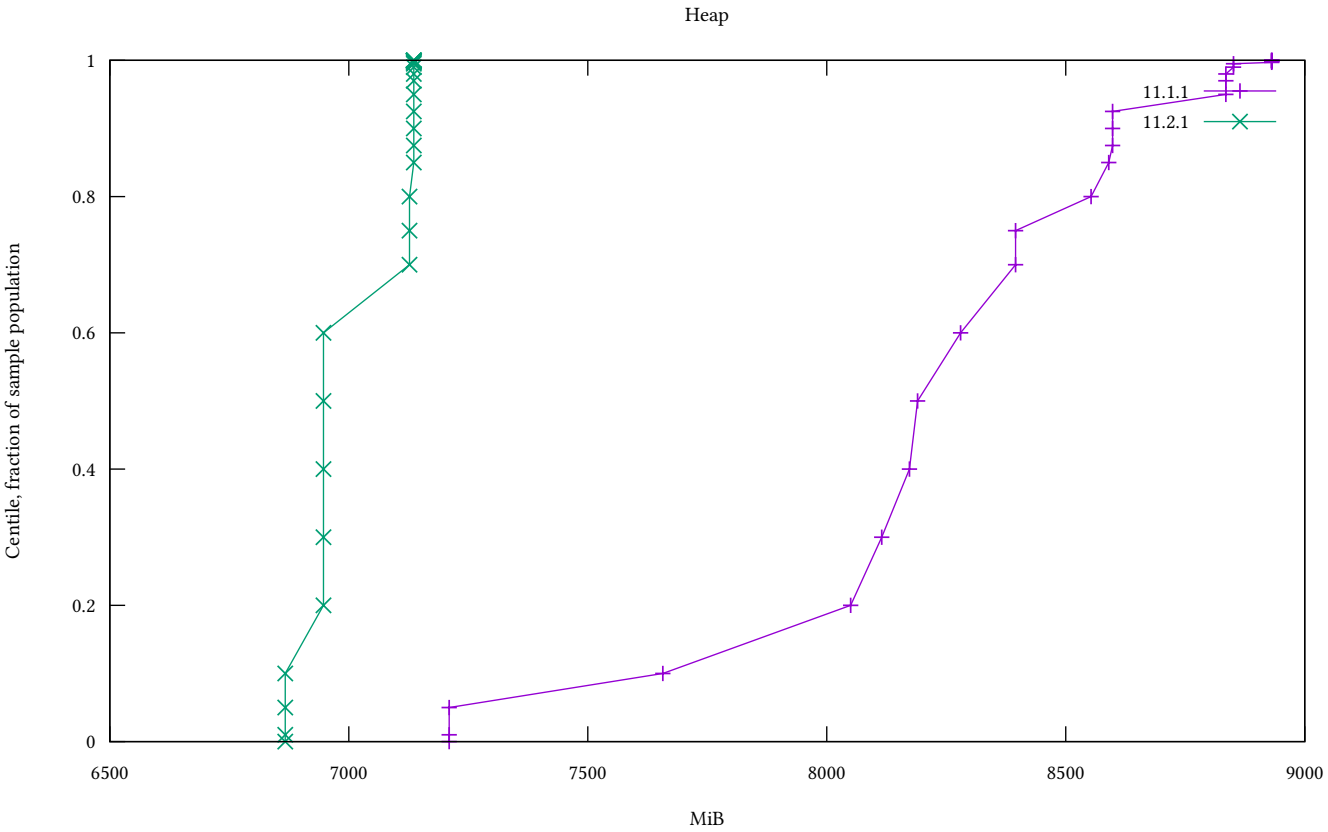


Figure 7: RTS heap size

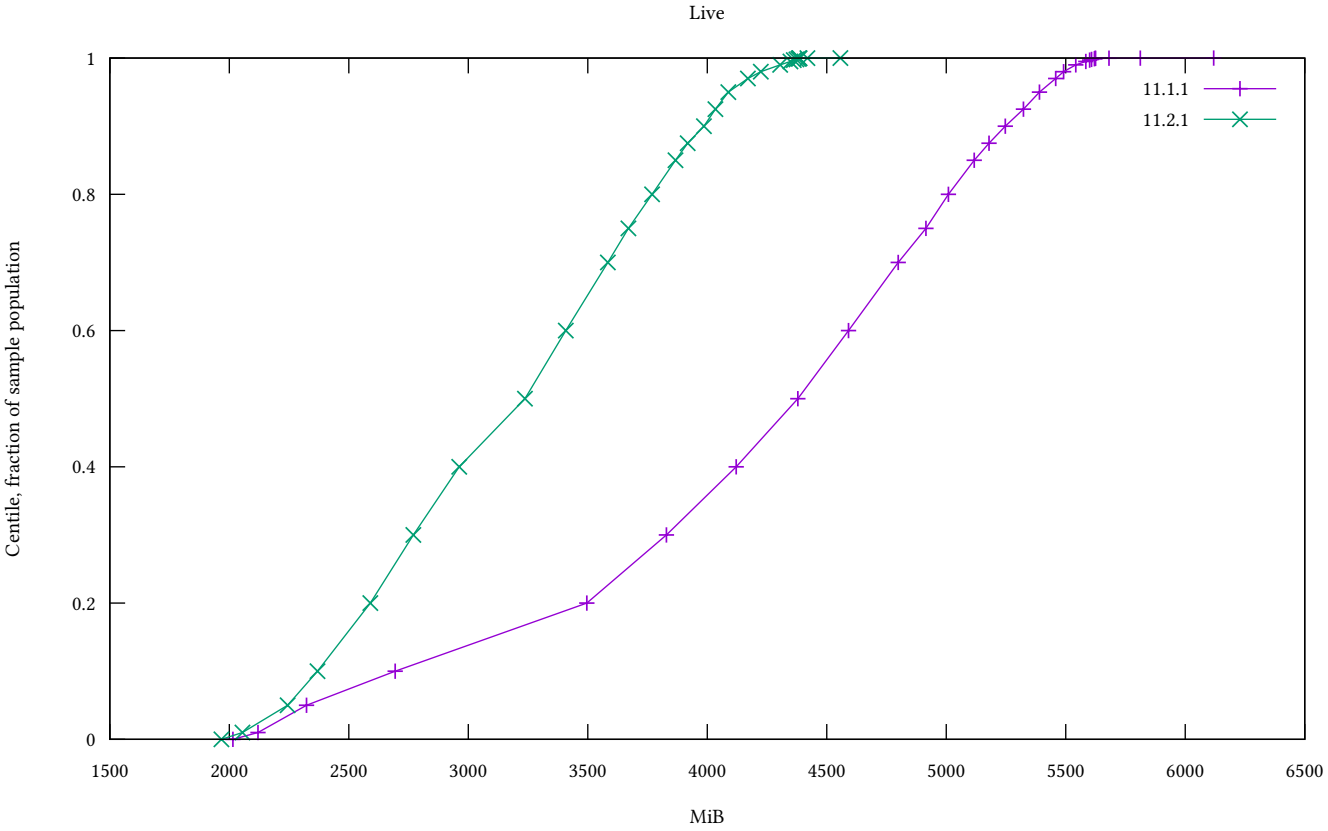


Figure 8: RTS live GC dataset

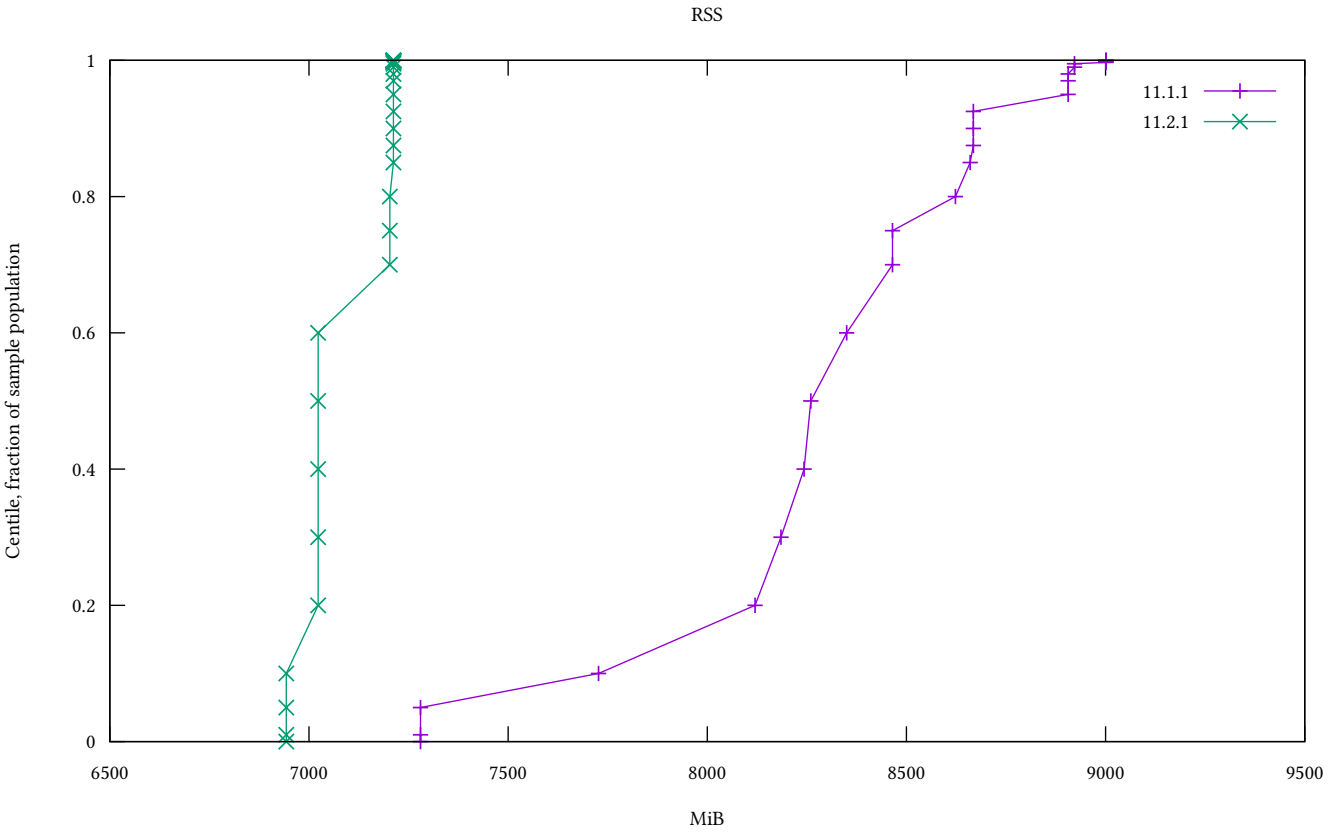


Figure 9: Kernel RSS

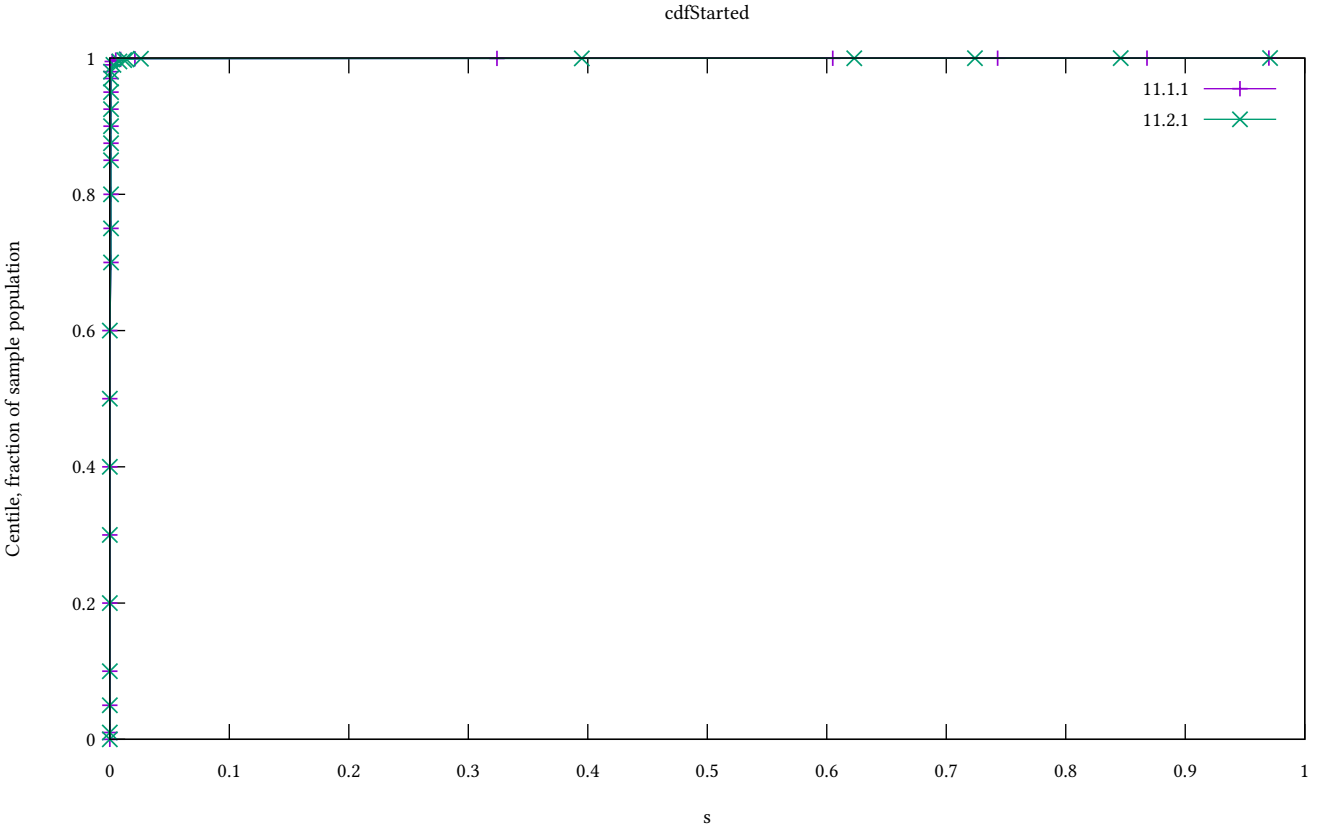


Figure 10: Forge loop tardiness

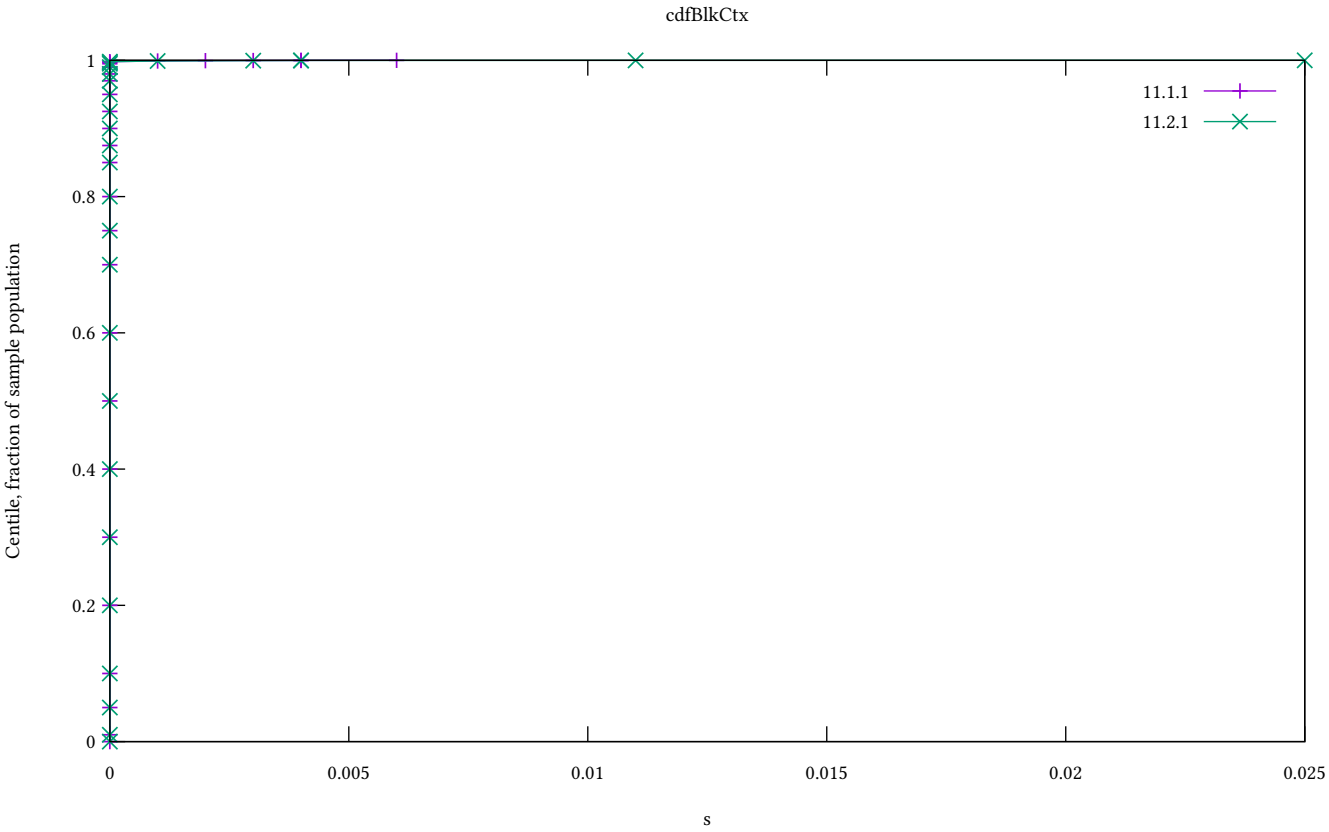


Figure 11: Block context acquisition delay

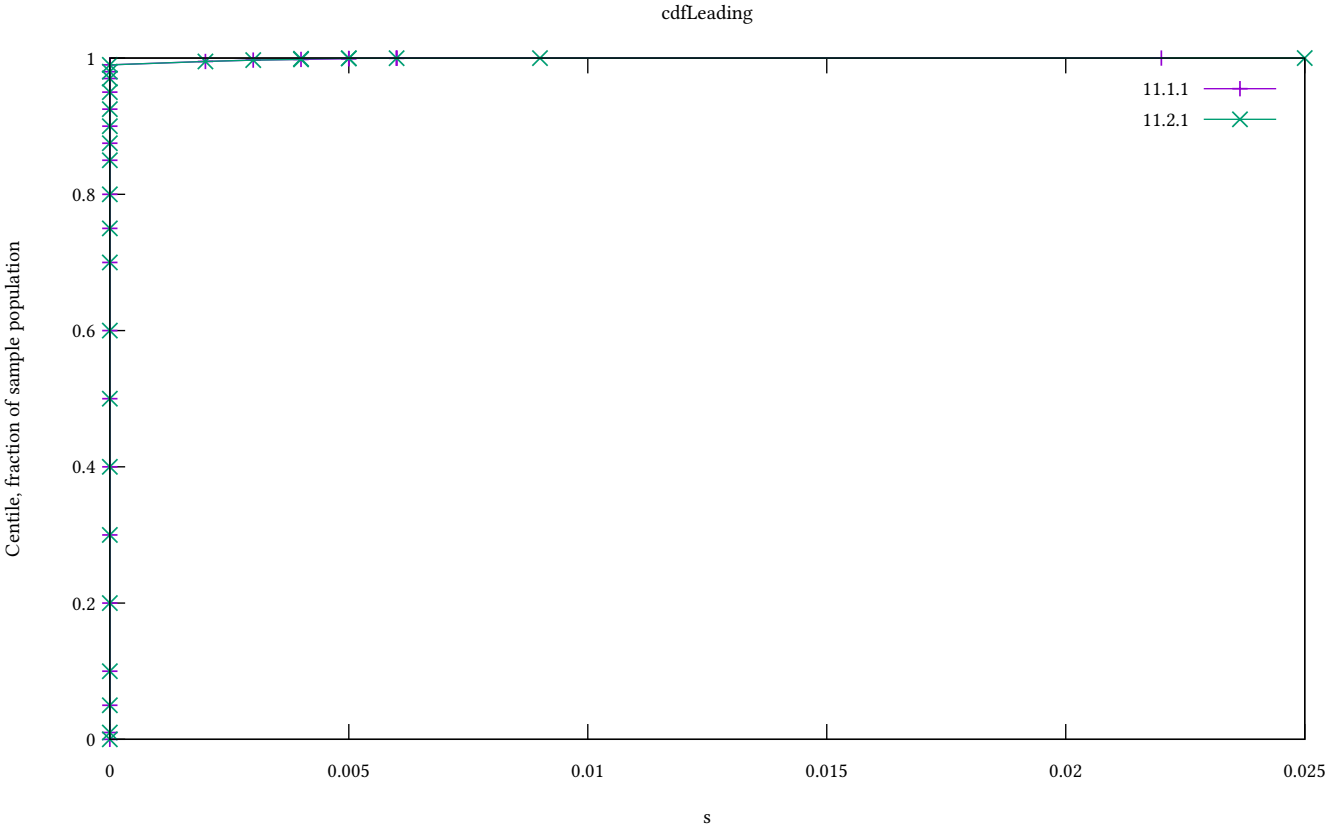


Figure 12: Leadership check duration

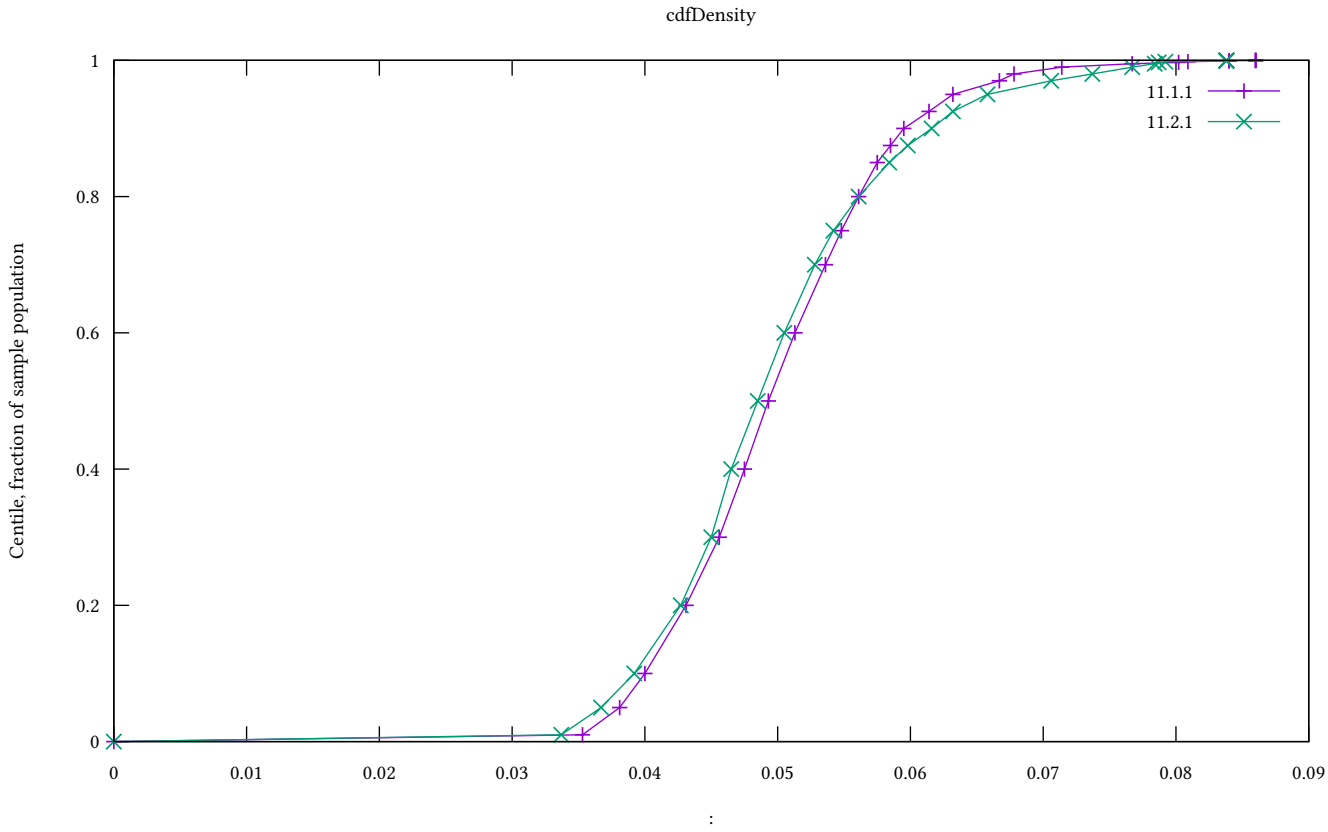


Figure 13: Chain density

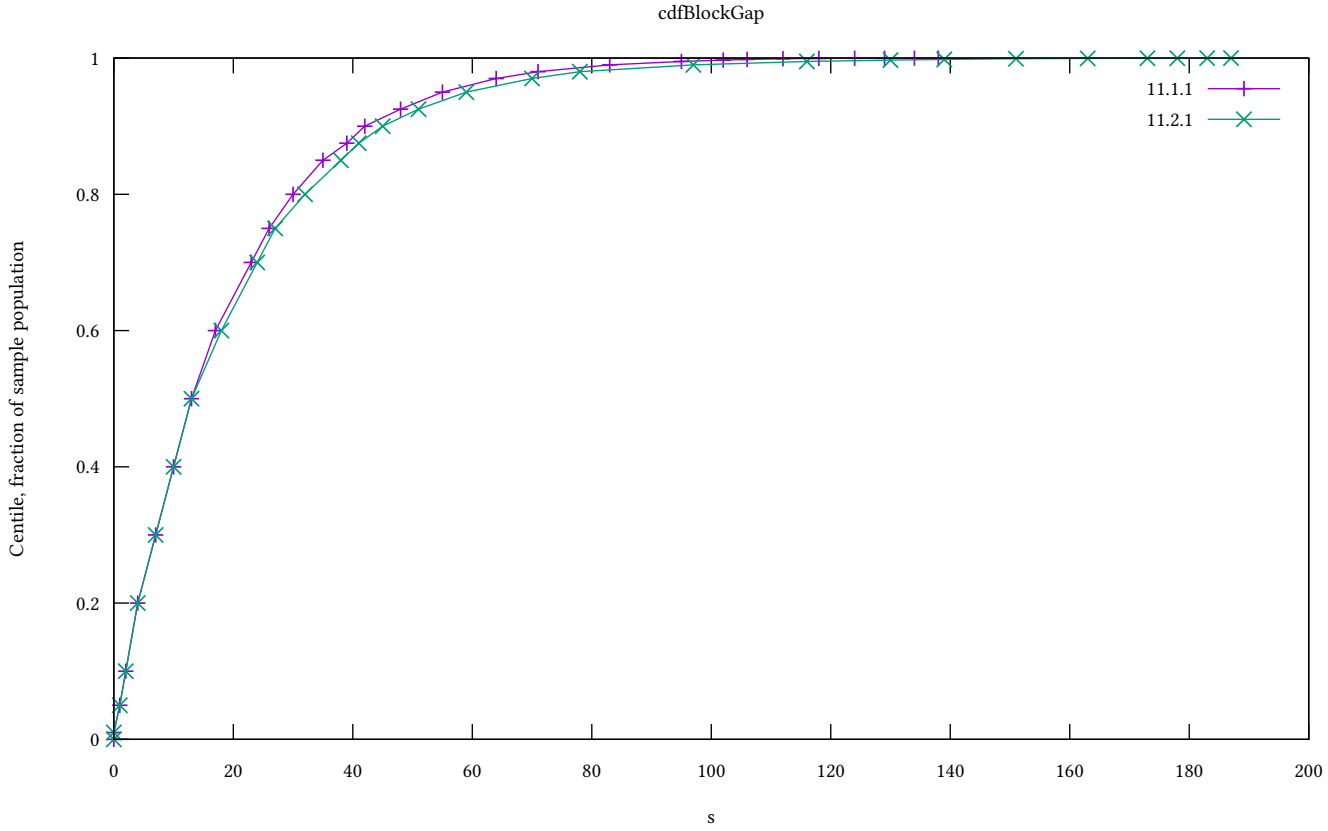


Figure 14: Interblock gap

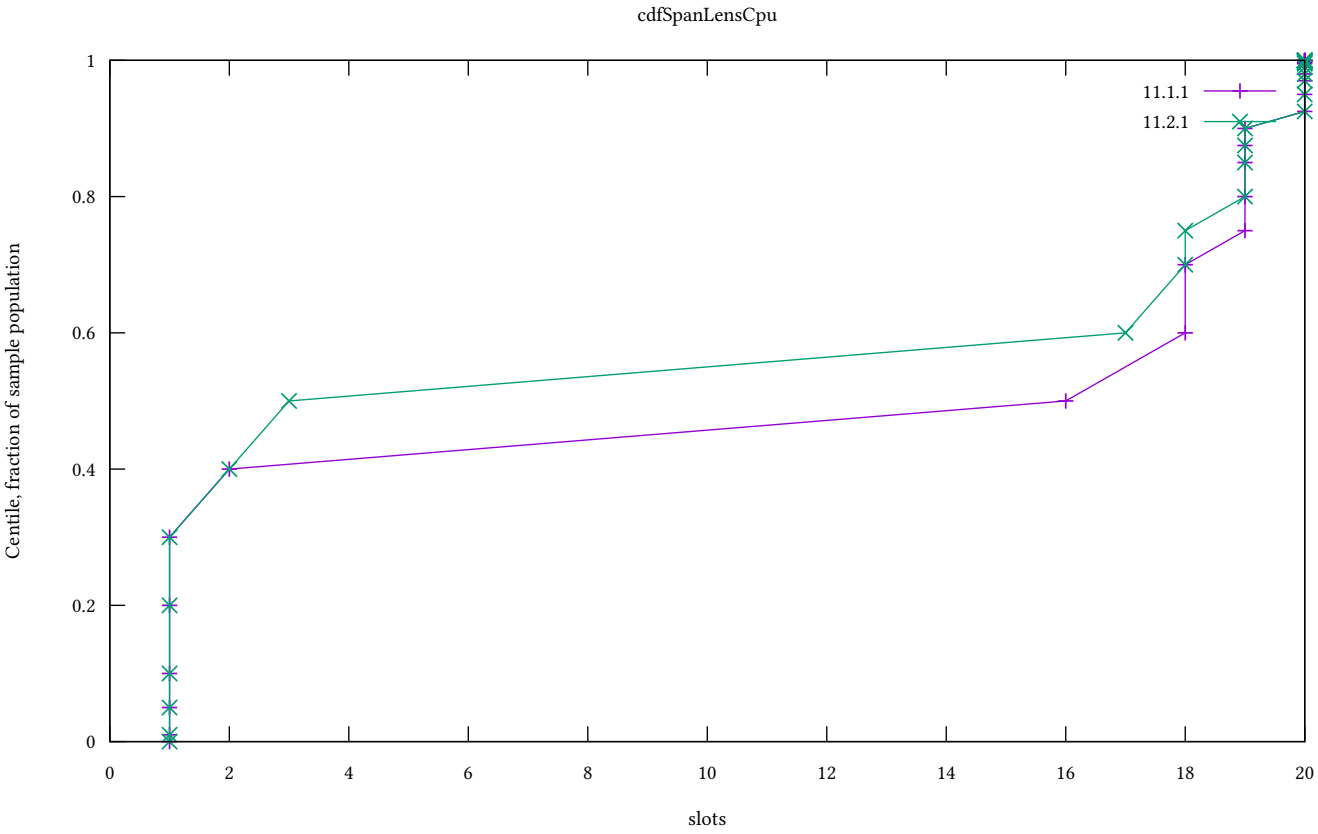


Figure 15: CPU 85% peaks, all

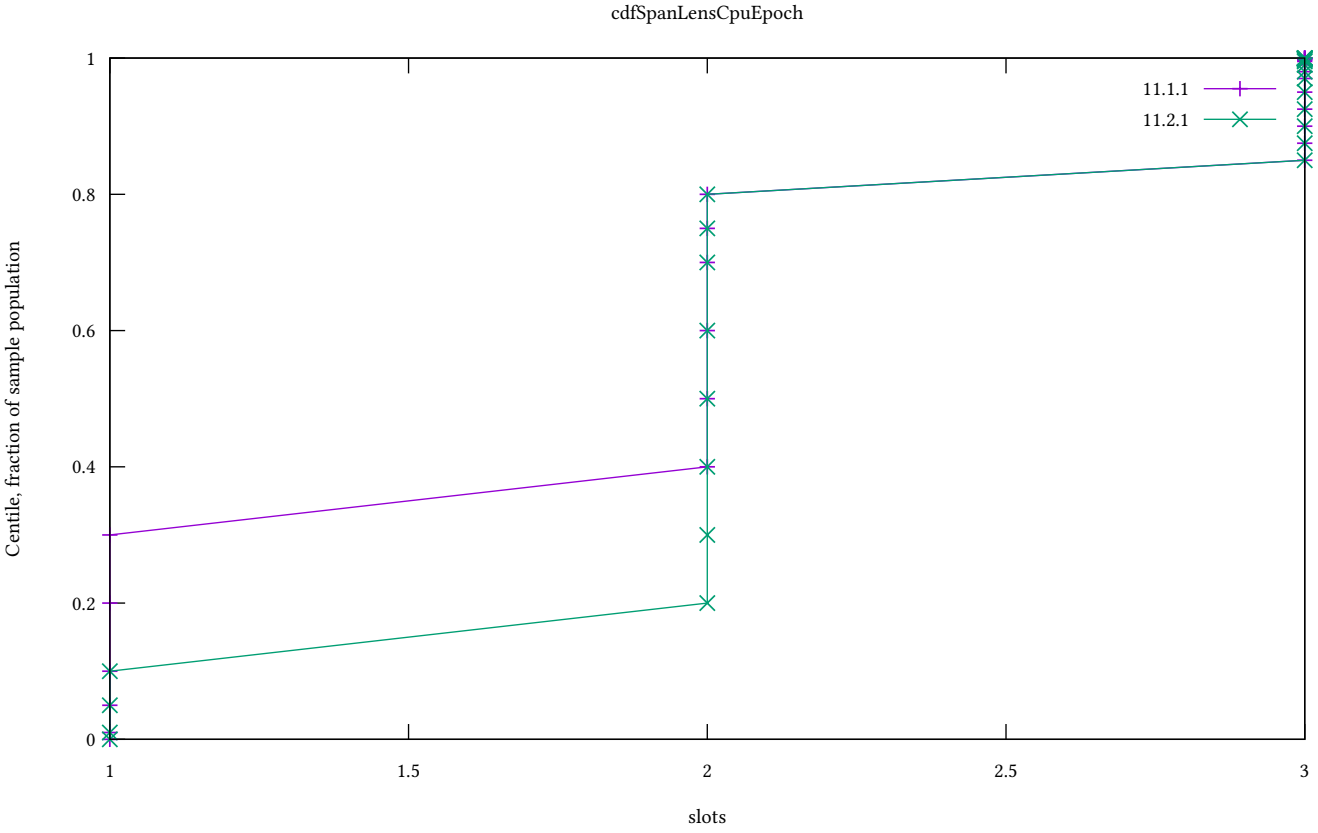


Figure 16: CPU 85% peaks, ep. trans.

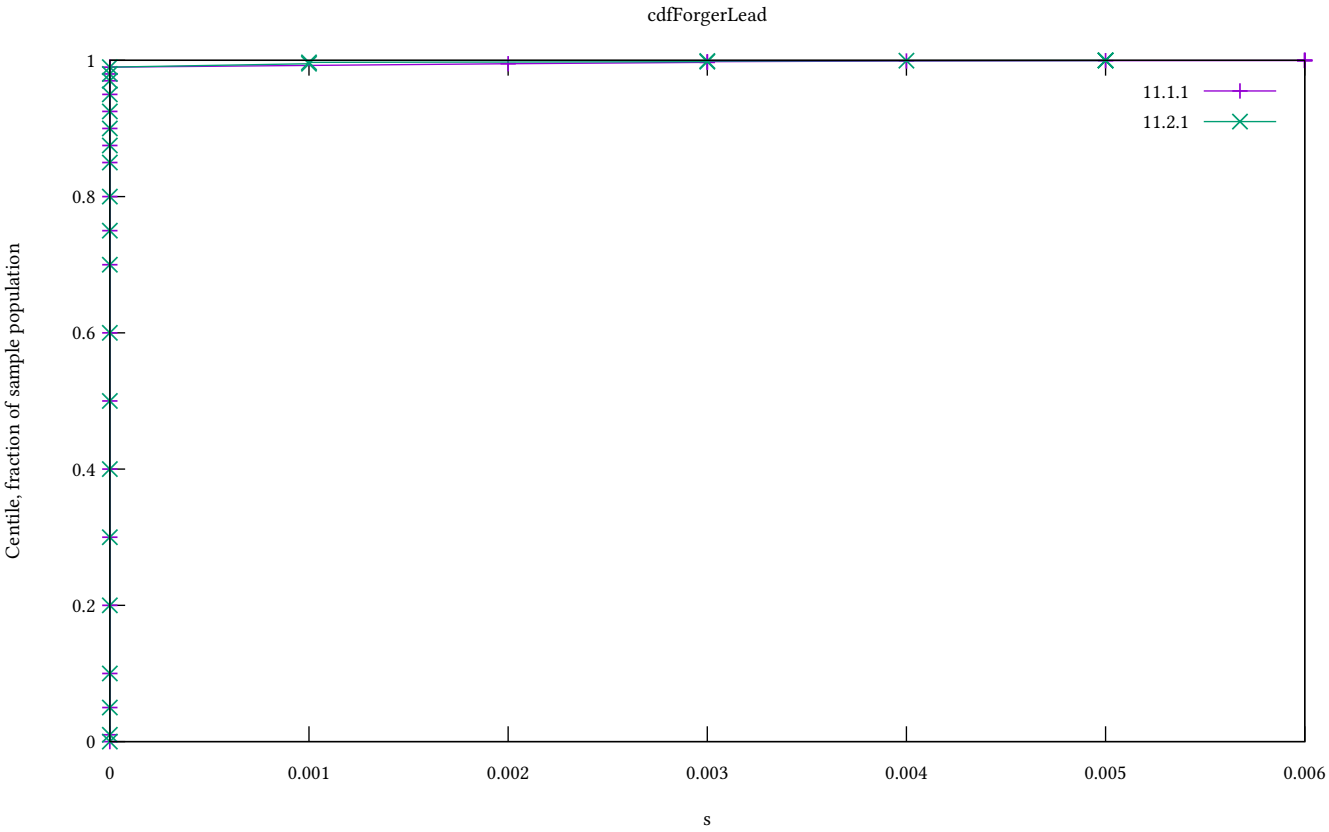


Figure 17: Leadership check duration

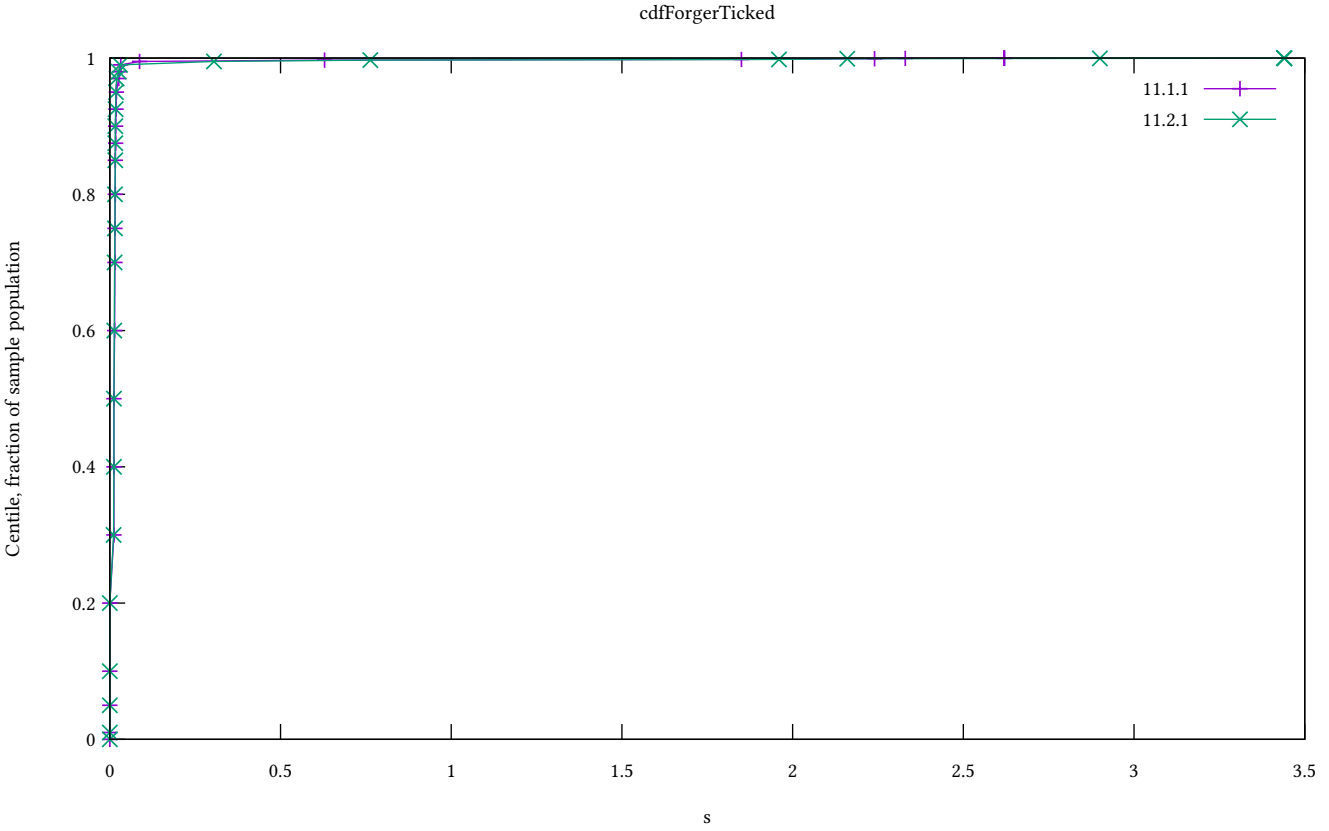


Figure 18: Ledger ticking

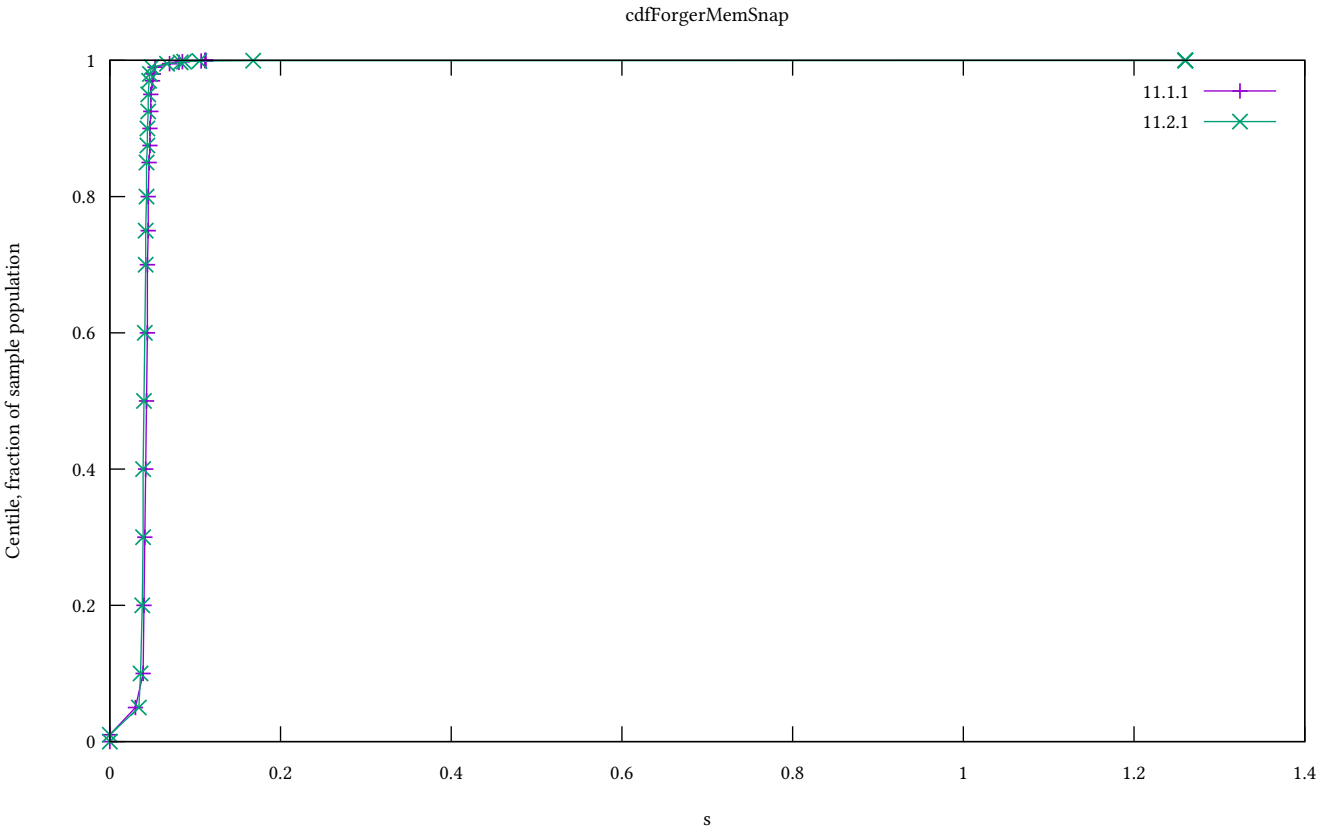


Figure 19: Mempool snapshotting

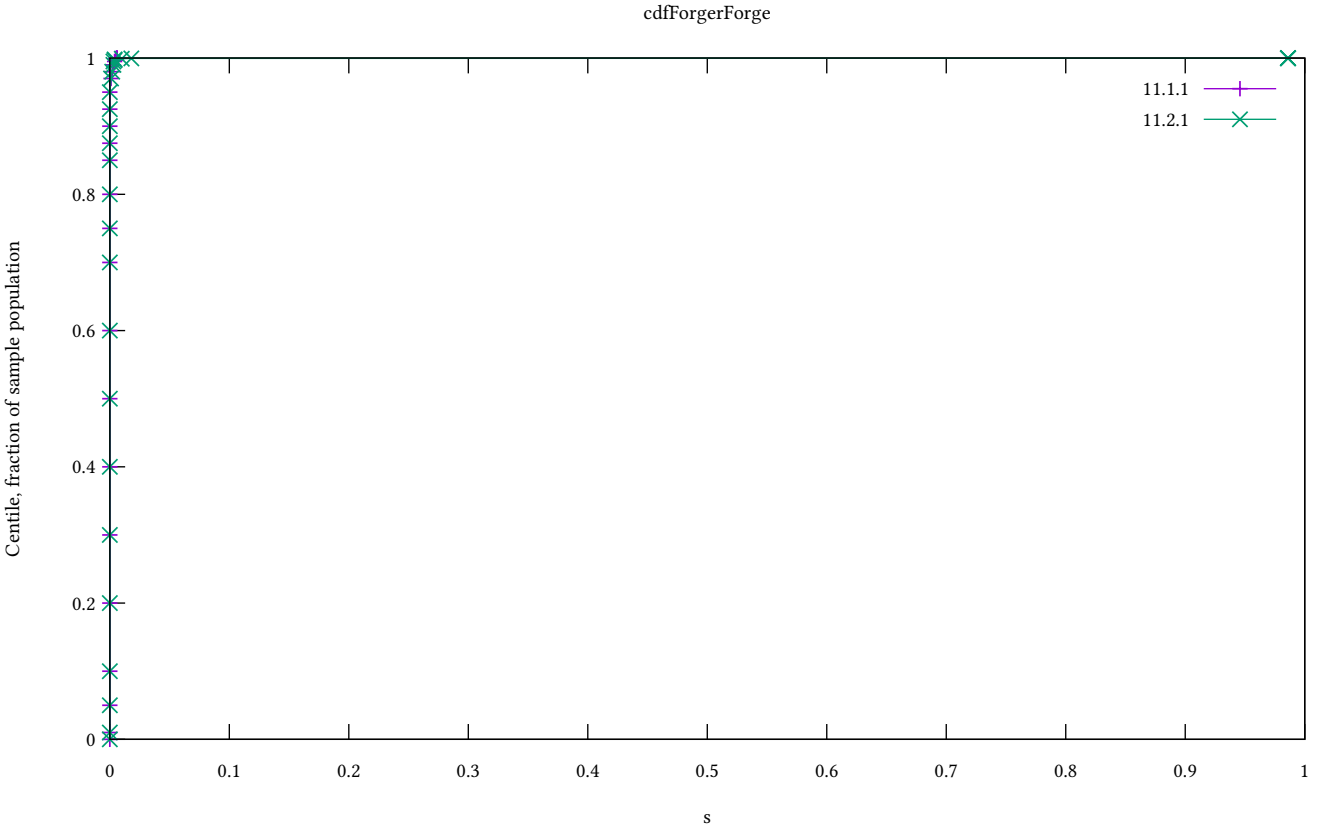


Figure 20: Leadership to forged

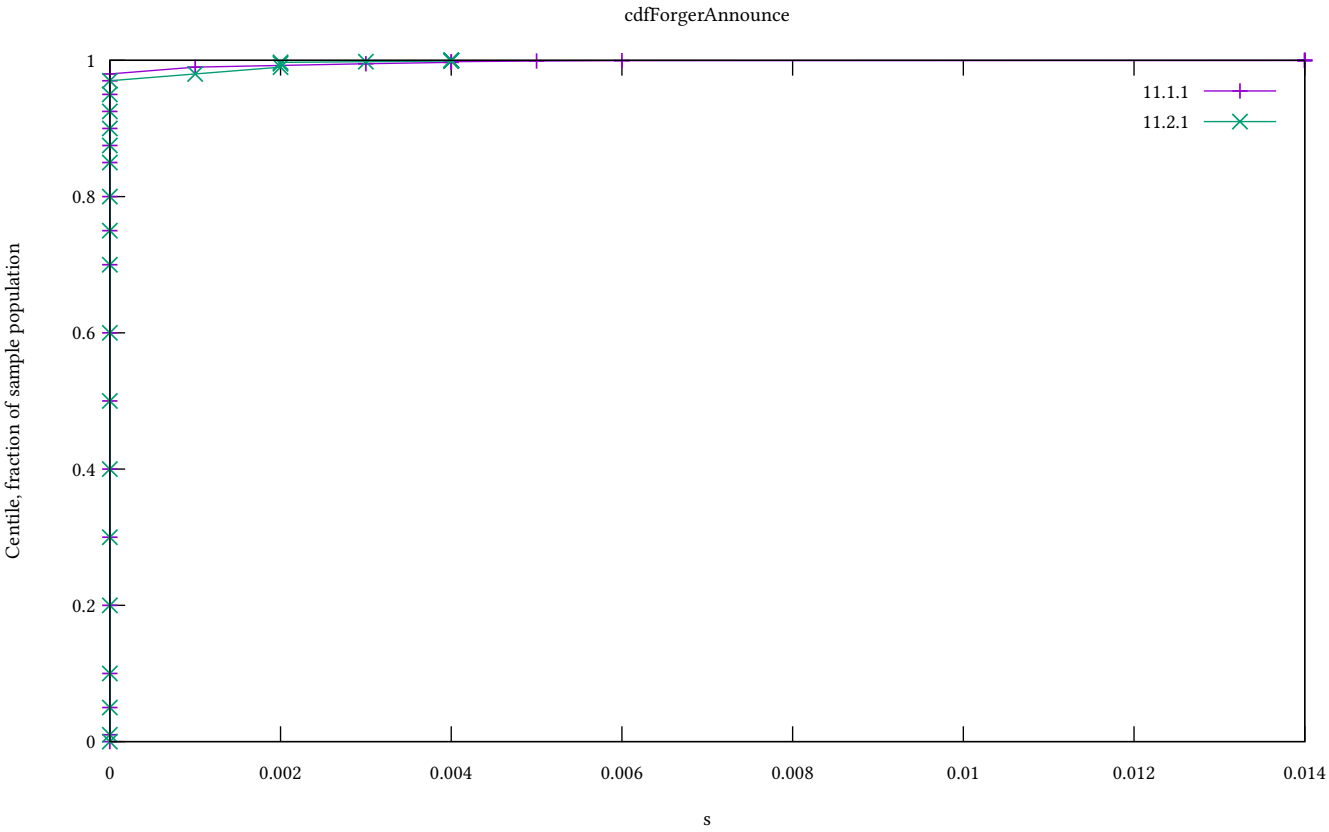


Figure 21: Forged to announced

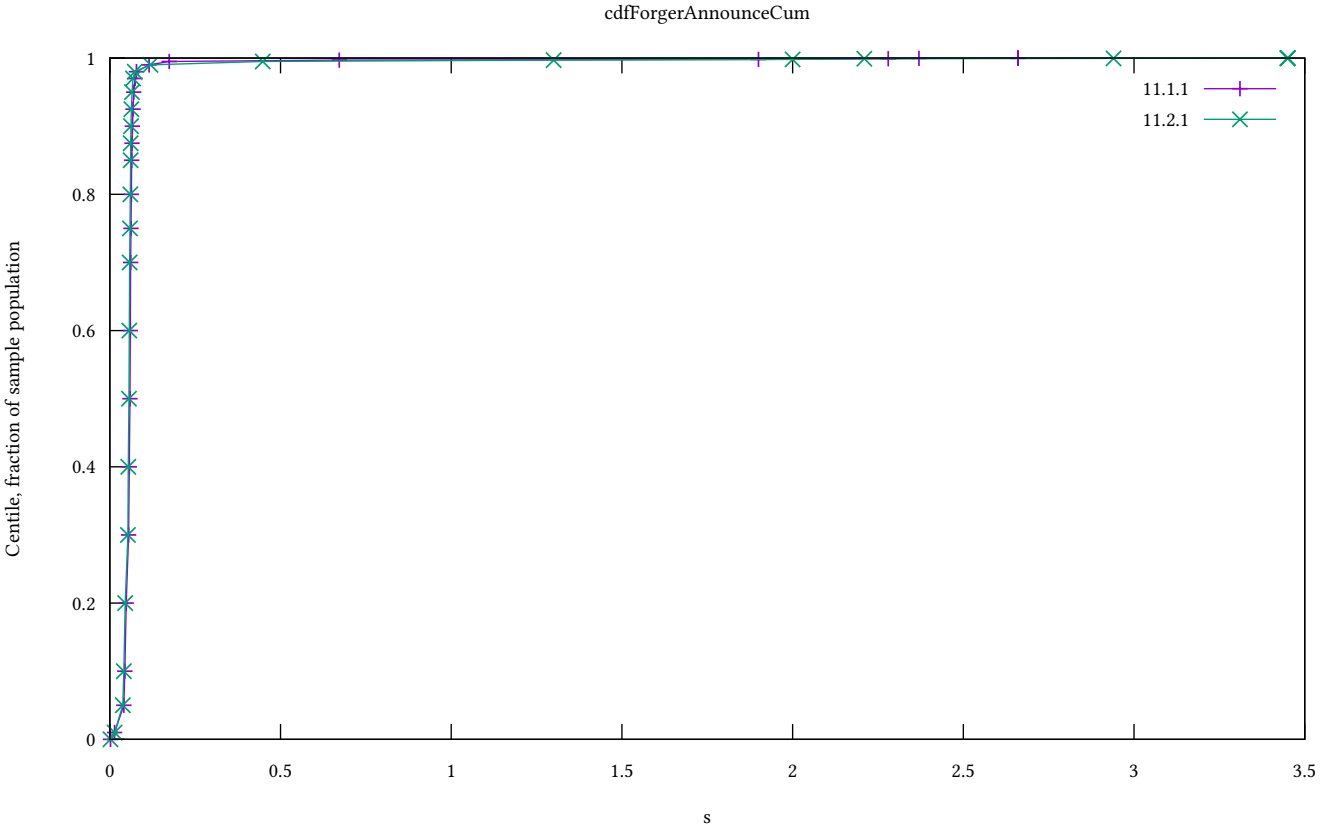


Figure 22: Slot start to announced

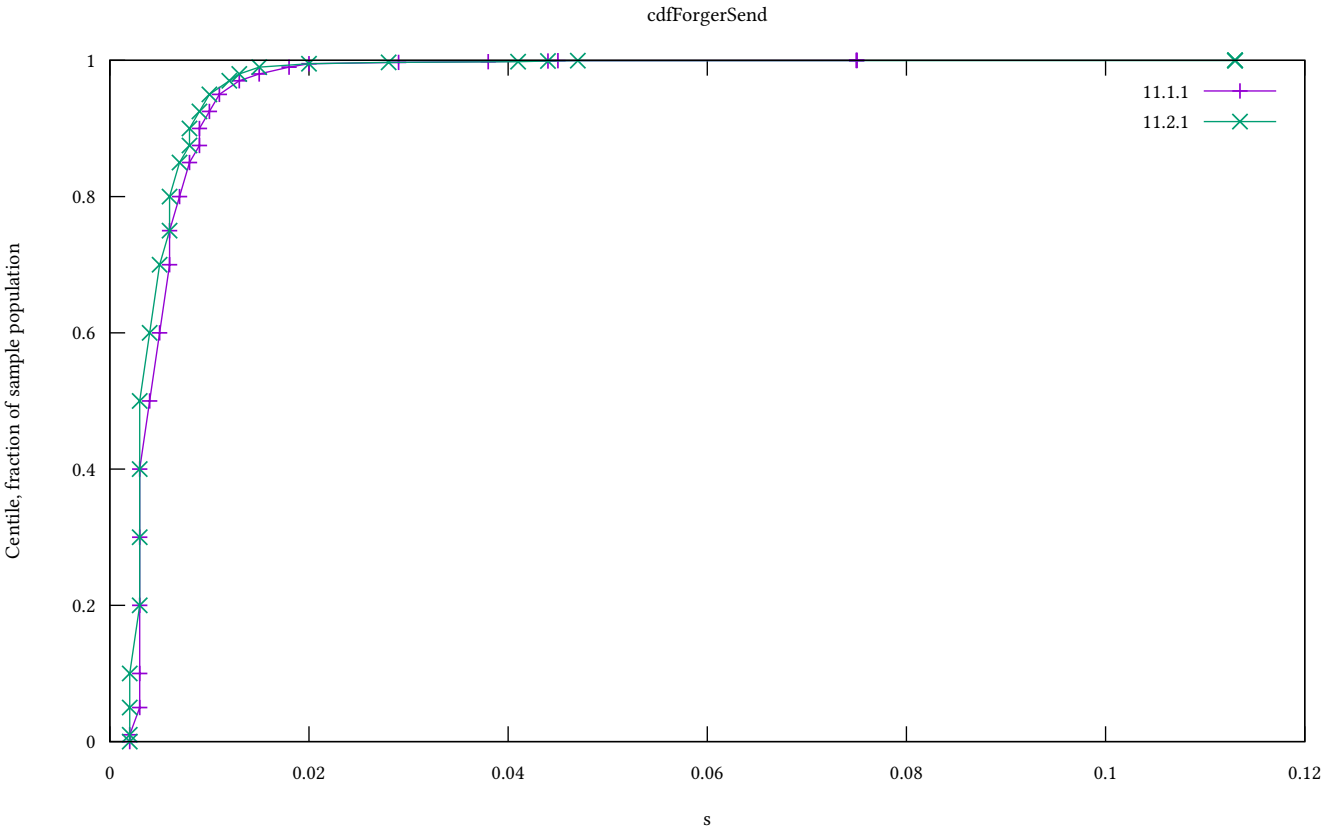


Figure 23: Forged to sending

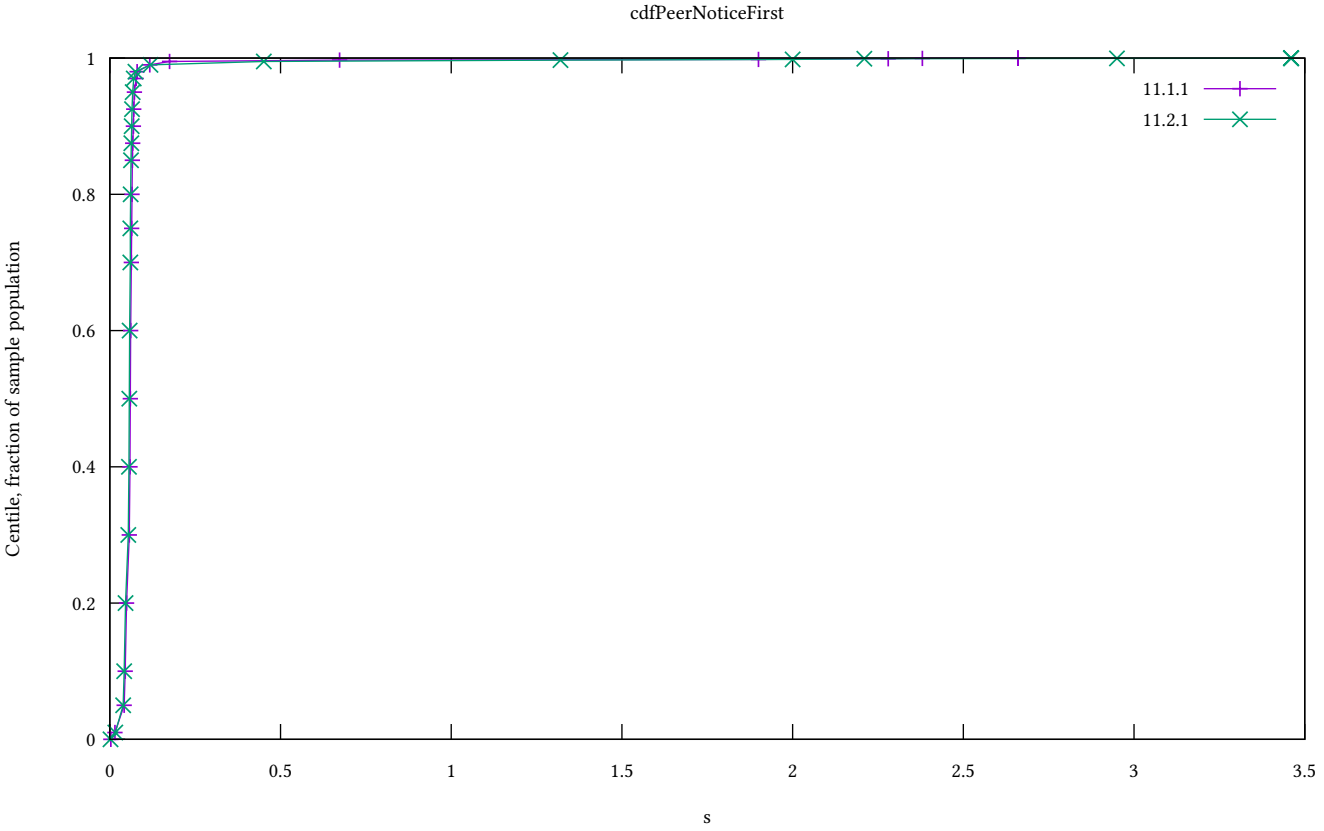


Figure 24: First peer notice

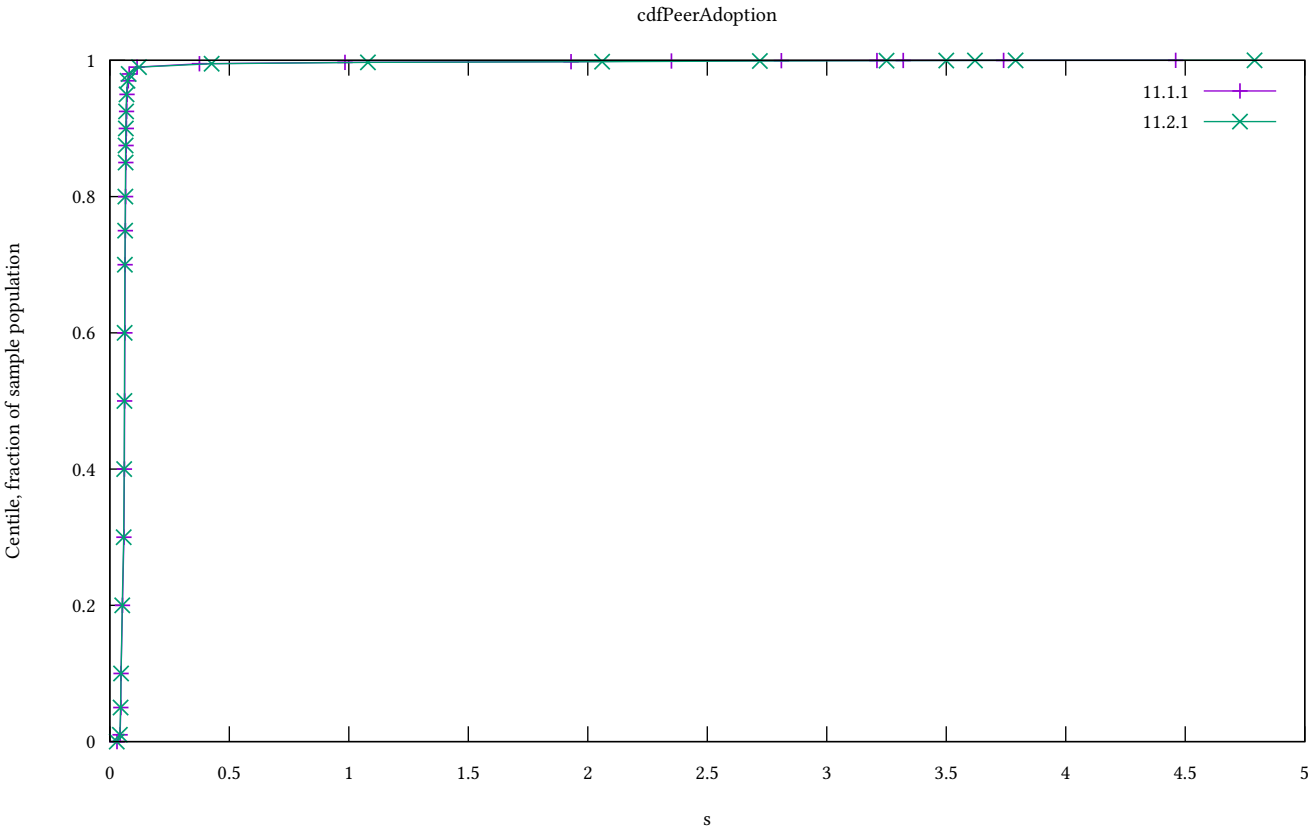


Figure 25: Fetched to adopted

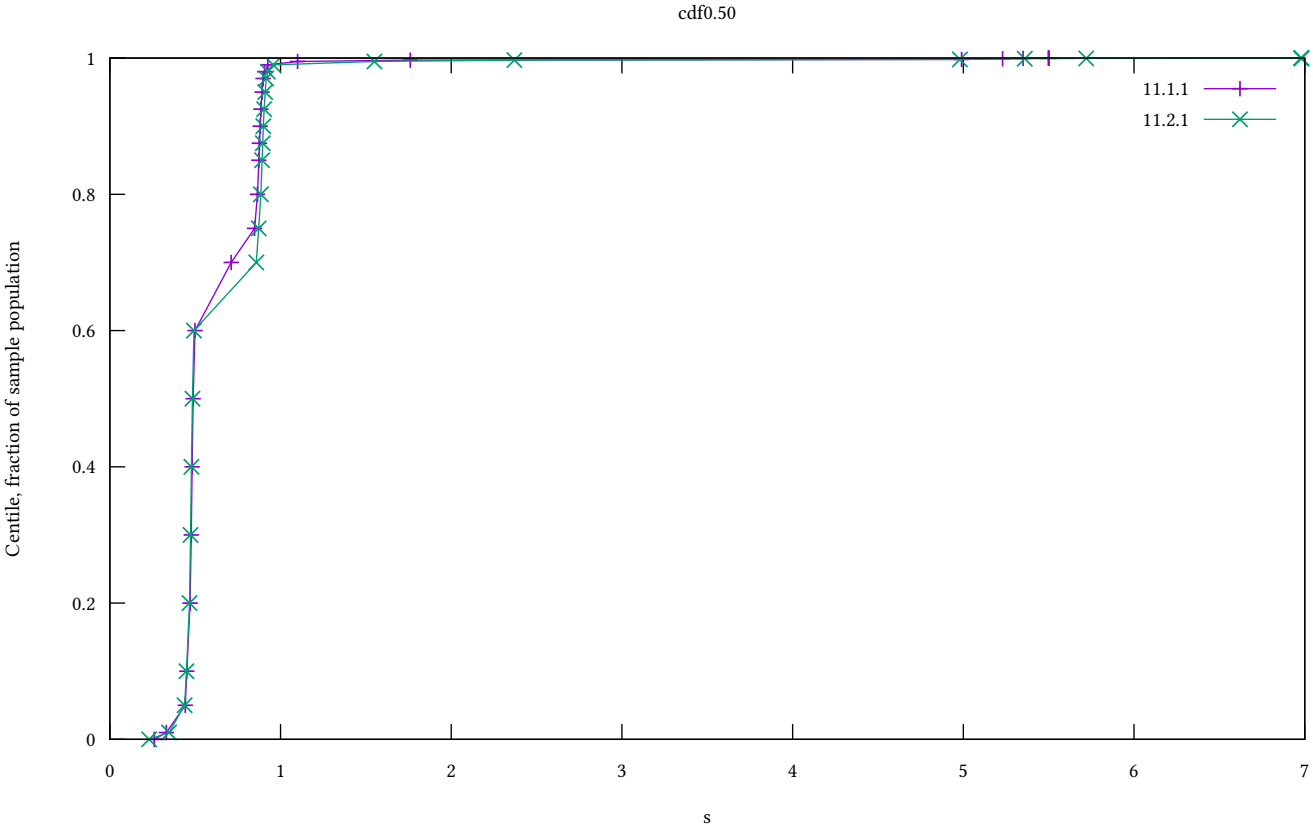


Figure 26: 0.50 adoption

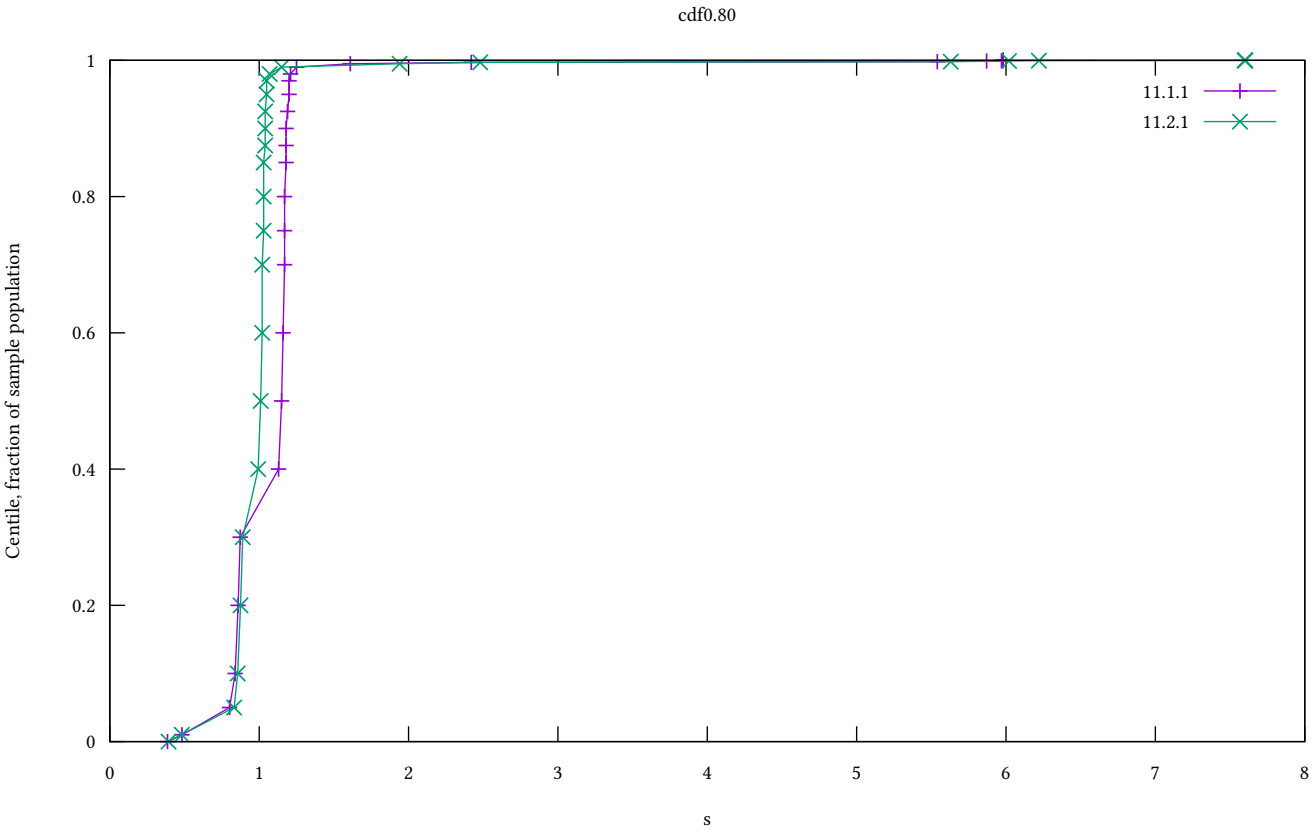


Figure 27: 0.80 adoption

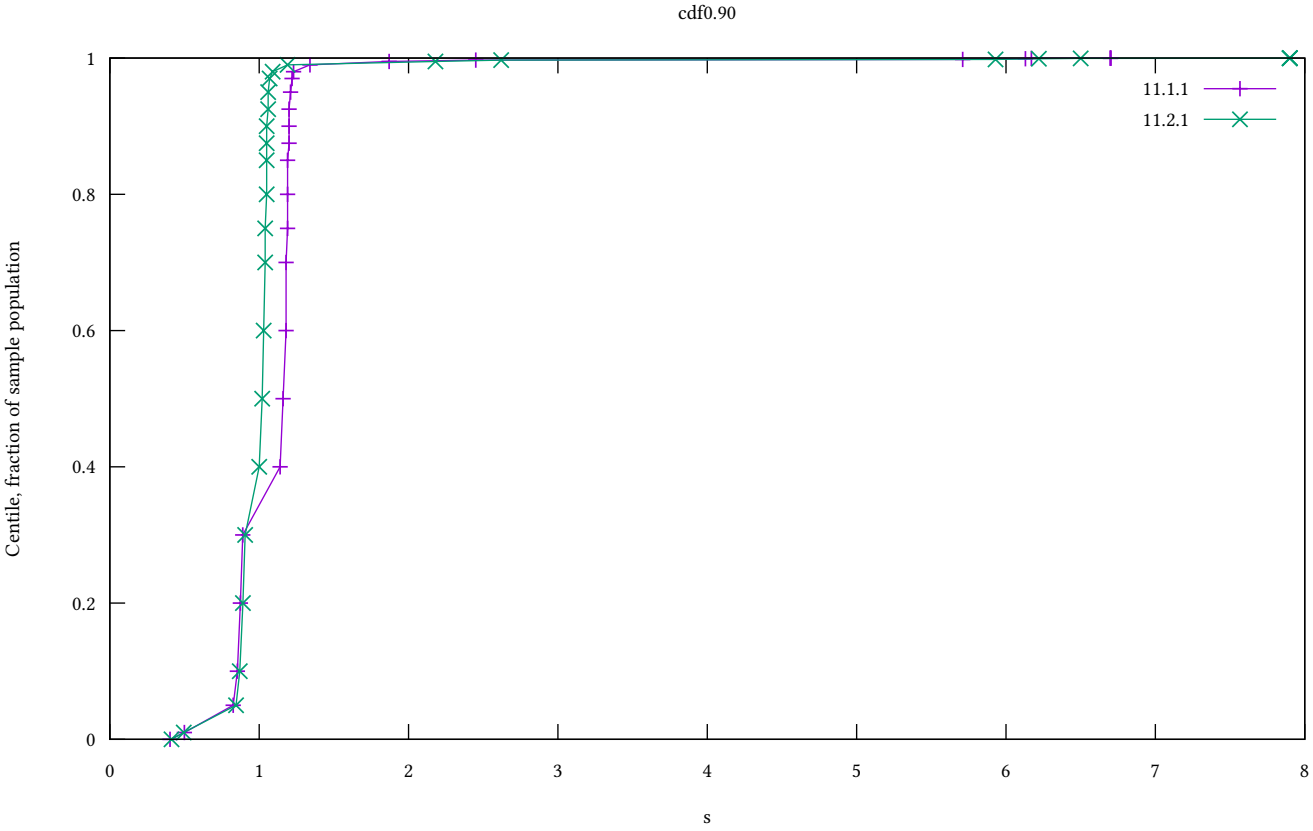


Figure 28: 0.90 adoption

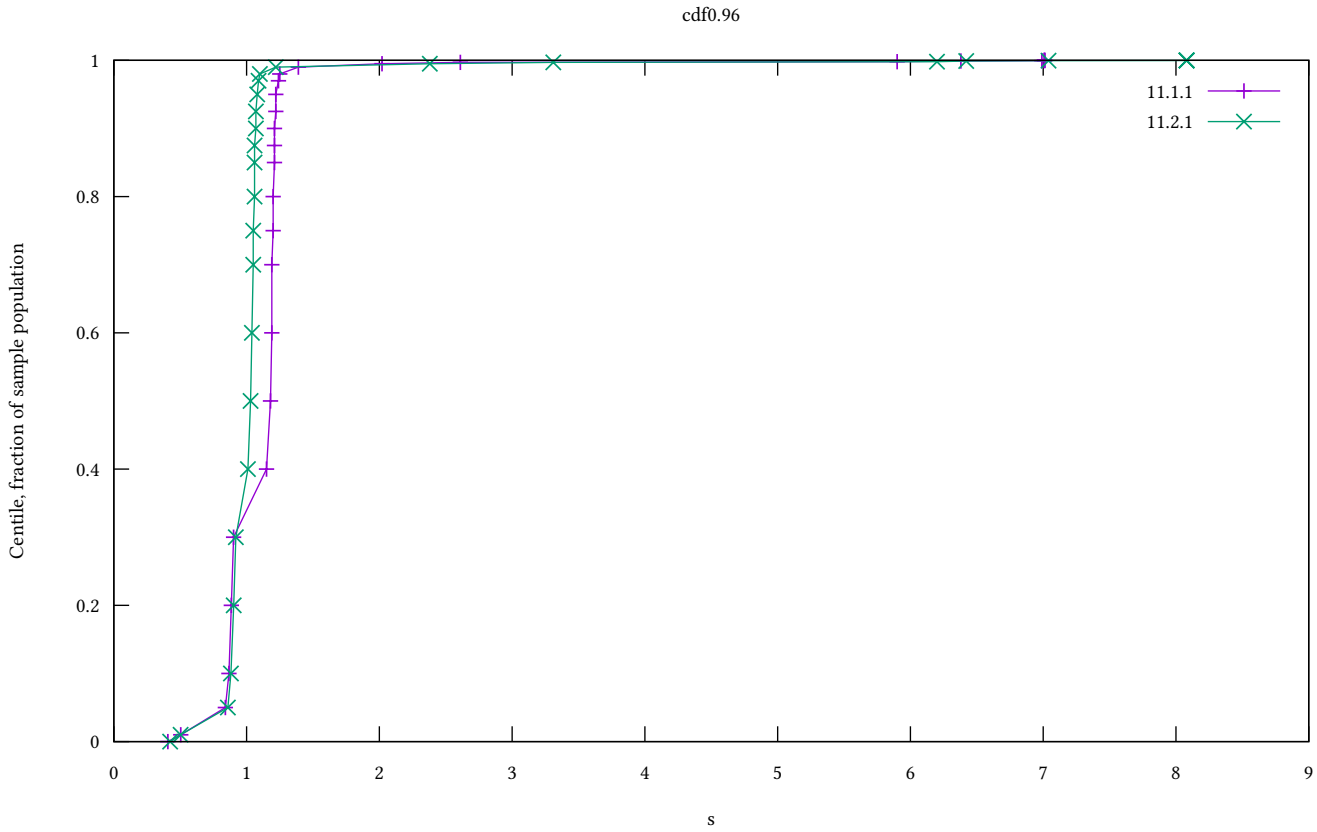


Figure 29: 0.96 adoption

Appendix B: data dictionary

Block propagation metrics

0.50 adoption (*cdf0.50*) – Time since slot start to block's adoption by 50% of the cluster.

0.80 adoption (*cdf0.80*) – Time since slot start to block's adoption by 80% of the cluster.

0.90 adoption (*cdf0.90*) – Time since slot start to block's adoption by 90% of the cluster.

0.92 adoption (*cdf0.92*) – Time since slot start to block's adoption by 92% of the cluster.

0.94 adoption (*cdf0.94*) – Time since slot start to block's adoption by 94% of the cluster.

0.96 adoption (*cdf0.96*) – Time since slot start to block's adoption by 96% of the cluster.

0.98 adoption (*cdf0.98*) – Time since slot start to block's adoption by 98% of the cluster.

1.00 adoption (*cdf1.00*) – Time since slot start to block's adoption by 100% of the cluster.

Block size (*cdfBlockSize*) – Block size, in bytes

Chained to forged blocks (*cdfBlocksChainedRatio*) – For each host, ratio of blocks that made into chain / all forged

Filtered to chained blocks (*cdfBlocksFilteredRatio*) – For each host, ratio of blocks that passed filtering / all on chain

Blocks per host (*cdfBlocksPerHost*) – For each host, number of blocks made during the entire observation period

Forged to self-adopted (*cdfForgerAdoption*) – Time between block forging completion and adoption (TraceAdoptedBlock)

Forged to announced (*cdfForgerAnnounce*) – Time between block forging completion and header announcement (ChainSyncServerEvent.TraceChainSyncServerRead.AddBlock)

Slot start to announced (*cdfForgerAnnounceCum*) – Time since slot start until header announcement (ChainSyncServerEvent.TraceChainSyncServerRead.AddBlock)

Acquired block context (*cdfForgerBlkCtx*) – Block context acquired (TraceBlockContext), relative to forge loop beginning

Leadership to forged (*cdfForgerForge*) – Time spent forging the block: TraceForgedBlock relative to positive leadership decision

Leadership check duration (*cdfForgerLead*) – Leadership check duration (TraceNodeIsNotLeader, TraceNodeIsLeader), relative to ledger view acquisition

Acquired ledger state (*cdfForgerLgrState*) – Ledger state acquired (TraceLedgerState), relative to block context acquisition

Acquired ledger view (*cdfForgerLgrView*) – Ledger view acquired (TraceLedgerView), relative to ledger state acquisition

Mempool snapshotting (*cdfForgerMemSnap*) – Time spent taking a mempool snapshot (TraceForgingMempoolSnapshot), relative to ledger ticking conclusion

Forged to sending (*cdfForgerSend*) – Time between block forging completion and begin-of-sending (TraceBlockFetchServerSendBlock)

Started forge loop iteration (*cdfForgerStart*) – Forge loop iteration delay (TraceStartLeadershipCheck), relative to slot start

Ledger ticking (*cdfForgerTicked*) – Time spent ticking the ledger state (TraceForgeTickedLedgerState), relative to leadership check completion

Height battles (*cdfHeightBattle*) – For a given block on chain, number of other blocks at the same height, forged for another slot

Fetched to adopted (*cdfPeerAdoption*) – Time until the peer adopts the block (TraceAddBlockEvent.AddedToCurrentChain), since it was fetched

Fetched to announced (*cdfPeerAnnounce*) – Time it took a peer to announce the block (ChainSyncServerEvent.TraceChainSyncServerUpdate), since it was fetched

Fetch duration (*cdfPeerFetch*) – Time it took the peer to complete fetching the block (BlockFetchClient.CompletedBlockFetch), after having requested it

First peer fetch (*cdfPeerFetchFirst*) – Time it took for the fastest peer to fetch the block (BlockFetchClient.CompletedBlockFetch), since block's slot start

First peer notice (*cdfPeerNoticeFirst*) – Time it took for the fastest peer to notice the block (ChainSyncClientEvent.TraceDownloadedHeader), since block's slot start

Notice to fetch request (*cdfPeerRequest*) – Time it took the peer to request the block body (BlockFetchClient.SendFetchRequest), after it have seen its header

Fetched to sending (*cdfPeerSend*) – Time until the peer started sending the block (BlockFetchServer.SendBlock), since it was fetched

Slot battles (*cdfSlotBattle*) – For a given block on chain, number of other blocks forged for the same slot, at any height

Cluster performance metrics

RTS alloc rate (*Alloc*) – RTS-reported allocation rate, MB/sec

Process CPU usage (*CentiCpu*) – Kernel-reported CPU process usage, % of a single core

RTS GC CPU usage (*CentiGC*) – RTS-reported GC CPU usage, % of a single core

RTS Mutator CPU usage (*CentiMut*) – RTS-reported mutator CPU usage, % of a single core

Filesystem reads (*FsRd*) – Number of bytes which this process really did cause to be fetched from the storage layer, per second

Filesystem writes (*FsWr*) – Number of bytes which this process caused to be sent to the storage layer, modulo truncate(), per second

Major GCs (*GcsMajor*) – Major garbage collection RTS events

Minor GCs (*GcsMinor*) – Minor garbage collection RTS events

RTS heap size (*Heap*) – RTS-reported heap size, MB

RTS live GC dataset (*Live*) – RTS-reported GC live data size, MB

Network reads (*NetRd*) – Network reads, kB/sec

Network writes (*NetWr*) – Network writes, kB/sec

Kernel RSS (*RSS*) – Kernel-reported RSS (Resident Set Size) of the process, MB

Block context acquisition delay (*cdfBlkCtx*) – Block context acquired (TraceBlockContext), relative to forge loop beginning

Interblock gap (*cdfBlockGap*) – Time between blocks

Proc CPU ÷ txns (*cdfCpuPerTx*) – Kernel-reported process CPU time, divided by the transactions processed. Not a per-tx cost: includes block validation, forging, snapshots and GC

Chain density (*cdfDensity*) – Block/slot ratio, for the last 'k' slots

GC CPU ÷ txns (*cdfGcPerTx*) – RTS-reported GC CPU time, divided by the transactions processed (as above). A heap regression shows here first

Leadership check duration (*cdfLeading*) – Leadership check duration (TraceNodeIsNotLeader, TraceNodeIsLeader), relative to ledger view acquisition

Ledger state acquisition delay (*cdfLgrState*) – Ledger state acquired (TraceLedgerState), relative to block context acquisition

Ledger view acquisition delay (*cdfLgrView*) – Ledger view acquired (TraceLedgerView), relative to ledger state acquisition

Live after majGC, during (*cdfLiveMajGcDuring*) – RTS-reported live data after a major GC that occurred during a ledger snapshot write (from its start until 5s after its completion)

Live after majGC, outside (*cdfLiveMajGcOutside*) – RTS-reported live data after a major GC that occurred outside of ledger snapshot writes. The decisive measure of retained data; usually few samples

Mempool fill level (*cdfMempoolTxs*) – Transactions in the mempool, per analysed slot (time-weighted). Its fill level, not its configured capacity

Mut CPU ÷ txns (*cdfMutPerTx*) – RTS-reported mutator CPU time, divided by the transactions processed (as above)

Duration (*cdfSnapDuration*) – Time it took to write a ledger snapshot to disk (ChainDB.LedgerEvent.Snapshot.TookSnapshot)

Taken (*cdfSnapTaken*) – Ledger snapshots taken (completed) per host, during the analysed slots. 0 most likely indicates a trace parsing failure

CPU 85% peaks, all (*cdfSpanLensCpu*) – Length of over-85% CPU usage peaks (consecutive slots), slots: all of them, whatever their cause

CPU 85% peaks, ep. trans. (*cdfSpanLensCpuEpoch*) – Length of the over-85% CPU usage peak ending at most 3 slots before the epoch's first block is adopted (the ledger's epoch transition), counted from the epoch's first slot, slots

CPU 85% peaks, other (*cdfSpanLensCpuOther*) – Length of over-85% CPU usage peaks neither at an epoch transition nor during a ledger snapshot write, slots. Unexplained saturation shows here

CPU 85% peaks, snapshot (*cdfSpanLensCpuSnap*) – Length of over-85% CPU usage peaks overlapping a ledger snapshot write (from its start until 5s after its completion), slots. Compared to Duration: CPU-bound vs. I/O-bound snapshots. Needs snapshot durations, as stored by locli 3.0

Forge loop tardiness (*cdfStarted*) – Forge loop iteration start delay (TraceStartLeadershipCheck), relative to slot start

Forge loop starts (*cdfStarts*) – For any given slot, how many forging loop starts were registered

Txns accepted (*cdfTxsAccepted*) – Processed transactions that were added to the mempool

Txns processed (*cdfTxsProcessed*) – Transactions received from peers and submitted to the mempool, per host, over the analysed slots

Reject ratio (*cdfTxsRejectRatio*) – Rejected / processed transactions. Explains differences between raw and per-transaction CPU usage

Txns rejected (*cdfTxsRejected*) – Processed transactions not added to the mempool: invalid, or already known. Run total of processed minus accepted, as the node's per-batch count isn't reliable