

AnalyzerChain Fetcher GetLinks This shows those three implementation changes. Fetcher GetLinks Fetcher chain.outQ chain.inQ GetLinks Fetcher CrawlStateManager Proxy to chain.outQ.put rec =chain.outQ.get() if rec is FetchRecord: mv urlQ aside and replace host = hostQ.get(max_priority=time()) load URLs from that hosts hostUrlQ urlQ.put(rec) fanout(accumulate(mergefiles())) if rec is HostRecord: add hosts to hostQ Chain.inQ.put(host) hostQ.put(rec) "urlQ" "hostQ" (BPQ) (BPQ) "hostUrlQ" "hostUrlQ" "hostUrlQ" (BPQ) (BPQ) (BPQ) •••

AnalyzerChain Fetcher GetLinks Notes on approaches to implementing the fanout. Fetcher Stats GetLinks Fetcher chain.outQ chain.inQ GetLinks **Fetcher** CrawlStateManager Proxy to CSM main while loop: chain.outQ.put rec =chain.outQ.get() Fanout Child: if rec is FetchRecord: sort urlQ files first on hostid, second on docid urlQ.put(rec) as accumulator iterates through these records: elif rec is HostRecord: if new hostid, open that hostUrlQ hostQ.put(rec) if new docid, hostURLQ.put(prev_rec) if chain.inQ.qsize() < MIN PACKED HOSTS: else: combine prev rec, next rec host = hostQ.get(max_priority=time()) load URLs from that hosts hostUrlQ if hostUrlQ is ready to sync: start child to merge-accumulate-sort hostUrlQ chain.inQ.put(host) if (hostQ is ready to sync): hostQ.sync() if (all hosts are not ready yet) or (urlQ is too big): os.rename(urlQ, urlQ syncing) "urlQ" (BPQ) open new urlQ (which will be empty) "hostQ" start child to merge-accumulate-fanout (BPQ) "hostUrlQ" "hostUrlQ" "hostUrlQ" (BPQ) (BPQ) (BPQ) •••