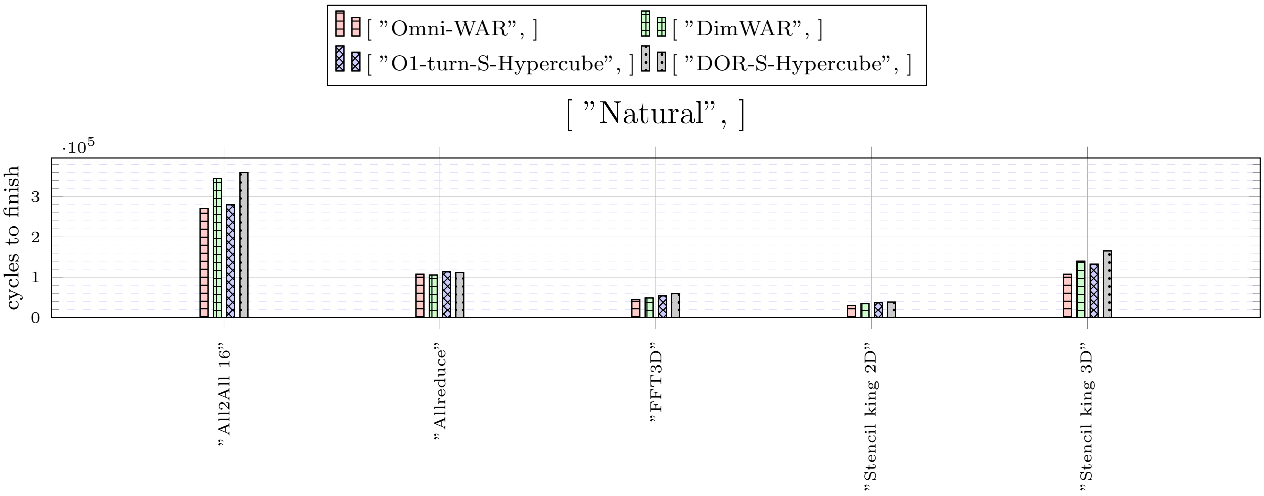

Figure 1: X: ["Natural",]

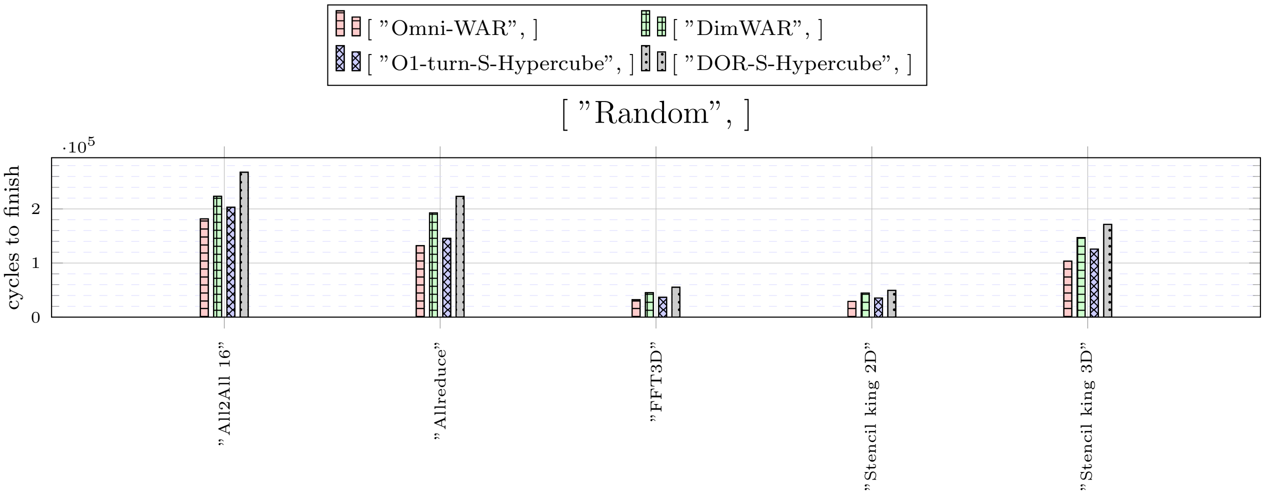

Figure 2: X: ["Random",]

The following versions used in the simulations.

- heads/alex-stable-release-TERA-1bbc361649a6952f6df662d4ea983c6a89d582f7(0.6.3)

```
Configuration{
  random_seed: ![ 1, 2, 3 ],
  warmup: 10000000000000,
  measured: 100000000000000,
  statistics_server_percentiles: [ 0, 5, 25, 50, 75, 95, 100 ],
  statistics_packet_percentiles: [ 0, 5, 25, 50, 75, 95, 100 ],
  general_frequency_divisor: 2,
  statistics_temporal_step: 1000,
  statistics_packet_definitions: [
    [
      [ ],
      [ ]]],
  topology: Hamming{
    servers_per_router: 8,
    sides: [ 8, 8 ]},
  traffic: TrafficMap{
    tasks: 512,
    map: ![
      Identity{ legend_name: "Natural" },
      RandomPermutation{ legend_name: "Random" }],
    application: ![
      All2All{ tasks: 512, data_size: 65536, legend_name: "All2All 16" },
      Stencil{
        tasks: 512,
        one_to_many_pattern: KingNeighbours{
          sides: [ 32, 16 ],
          distance: 1},
        message_size: 1024,
        rounds: 1,
        legend_name: "Stencil king 2D"},
      Stencil{
        tasks: 512,
        one_to_many_pattern: KingNeighbours{
          sides: [ 8, 8, 8 ],
          distance: 1},
        message_size: 1024,
        rounds: 1,
        legend_name: "Stencil king 3D"},
      All2AllLinear{
        task_space: [ 32, 16 ],
        message_size: 256,
        legend_name: "FFT3D"},
      AllReduce{ tasks: 512, data_size: 16384, algorithm: Hypercube, legend_name: "Allreduce" }]},
  router: InputOutput{
    virtual_channels: mecanismo![ 4, 2, 2, 1 ],
    virtual_channel_policies: mecanismo![
      [
        WideHops{ width: 1 },
        VecLabel{
          label_vector: [ 0, 64 ]},
        OccupancyFunction{ label_coefficient: 1, occupancy_coefficient: 1, product_coefficient: 0, constant_coefficient: 0, use_internal_space: true,
use_neighbour_space: true, aggregate: true },
        OccupancyFunction{ label_coefficient: 1, occupancy_coefficient: 1, product_coefficient: 0, constant_coefficient: 0, use_internal_space: true,
use_neighbour_space: true, aggregate: false },
        LowestLabel,
        EnforceFlowControl,
        Random],
      [
        MapLabel{
          label_to_policy: [
            ArgumentVC{
              allowed: [ 0 ]},
            ArgumentVC{
              allowed: [ 0 ]},
            ArgumentVC{
              allowed: [ 1 ]}]},
        VecLabel{
          label_vector: [ 0, 64, 0 ]},
        OccupancyFunction{ label_coefficient: 1, occupancy_coefficient: 1, product_coefficient: 0, constant_coefficient: 0, use_internal_space: true,
use_neighbour_space: true, aggregate: true },
        OccupancyFunction{ label_coefficient: 1, occupancy_coefficient: 1, product_coefficient: 0, constant_coefficient: 0, use_internal_space: true,
use_neighbour_space: true, aggregate: false },
        LowestLabel,
        EnforceFlowControl,
        Random],
      [
        VecLabel{
          label_vector: [ 0, 64, 64 ]},
        OccupancyFunction{ label_coefficient: 1, occupancy_coefficient: 1, product_coefficient: 0, constant_coefficient: 0, use_internal_space: true,
use_neighbour_space: true, aggregate: true },
        OccupancyFunction{ label_coefficient: 1, occupancy_coefficient: 1, product_coefficient: 0, constant_coefficient: 0, use_internal_space: true,
use_neighbour_space: true, aggregate: false },
        LowestLabel,
        EnforceFlowControl,
        Random],
      [
        VecLabel{
          label_vector: [ 0, 64, 64 ]},
        OccupancyFunction{ label_coefficient: 1, occupancy_coefficient: 1, product_coefficient: 0, constant_coefficient: 0, use_internal_space: true,
use_neighbour_space: true, aggregate: true },
        OccupancyFunction{ label_coefficient: 1, occupancy_coefficient: 1, product_coefficient: 0, constant_coefficient: 0, use_internal_space: true,
use_neighbour_space: true, aggregate: false },
        LowestLabel,
        EnforceFlowControl,
        Random]],
    allocator: Random,
    buffer_size: 160,
    bubble: false,
    flit_size: 16,
    intransit_priority: false,
    allow_request_busy_port: true,
    output_buffer_size: 80,
    crossbar_frequency_divisor: 1,
    crossbar_delay: 2},
  maximum_packet_size: 16,
  routing: mecanismo![
    OmniDimensionalDeroute{ allowed_deroutes: 2, include_labels: true, legend_name: "Omni-WAR" },
    DimWAR{
      order: [ 0, 1 ],
      legend_name: "DimWAR"},
    ChannelsPerHopPerLinkClass{
      use_total_hops: true,
      channels: [
        [
          [ 0 ],
          [ 0, 1 ]],
        [
          [ 1 ],
          [ 0, 1 ]],
        [
          [ 0, 1 ],
          [ 0, 1 ],
          [ 0, 1 ]]],
      default_channels: [ 0, 1 ],
      routing: Sum{
        policy: TryBoth,
        first_routing: GeneralDOR{
          region_logical_topology: [
            Hamming{
              servers_per_router: 8,
              sides: [ 8 ]},
            Hamming{
              servers_per_router: 8,
              sides: [ 8 ]}],
          routings: [
            SubTopologyRouting{
              logical_topology: Hamming{
                servers_per_router: 2,
                sides: [ 2, 2, 2 ]},
              map: Identity,
              logical_routing: DOR{
                order: [ 0, 1, 2 ]},
              livelock_avoidance: true,
              opportunistic_hops: true},
            SubTopologyRouting{
              logical_topology: Hamming{
                servers_per_router: 2,
                sides: [ 2, 2, 2 ]},
              map: Identity,
              logical_routing: DOR{
                order: [ 0, 1, 2 ]},
              opportunistic_hops: true,
              livelock_avoidance: true}]},
        second_routing: GeneralDOR{
          order: [ 1, 0 ],
          region_logical_topology: [
            Hamming{
              servers_per_router: 8,
              sides: [ 8 ]},
            Hamming{
              servers_per_router: 8,
              sides: [ 8 ]}],
          routings: [
            SubTopologyRouting{
              logical_topology: Hamming{
                servers_per_router: 2,
                sides: [ 2, 2, 2 ]},
              map: Identity,
              logical_routing: DOR{
                order: [ 0, 1, 2 ]},
              livelock_avoidance: true,
              opportunistic_hops: true},
            SubTopologyRouting{
              logical_topology: Hamming{
                servers_per_router: 2,
                sides: [ 2, 2, 2 ]},
              map: Identity,
              logical_routing: DOR{
                order: [ 0, 1, 2 ]},
              opportunistic_hops: true,
              livelock_avoidance: true}]},
        first_allowed_virtual_channels: [ 0 ],
        second_allowed_virtual_channels: [ 1 ]},
      legend_name: "O1-turn-S-Hypercube"},
    GeneralDOR{
      region_logical_topology: [
        Hamming{
          servers_per_router: 8,
          sides: [ 8 ]},
        Hamming{
          servers_per_router: 8,
          sides: [ 8 ]}],
      routings: [
        SubTopologyRouting{
          logical_topology: Hamming{
            servers_per_router: 2,
            sides: [ 2, 2, 2 ]},
          map: Identity,
          logical_routing: DOR{
            order: [ 0, 1, 2 ]},
          livelock_avoidance: true,
          opportunistic_hops: true},
        SubTopologyRouting{
          logical_topology: Hamming{
            servers_per_router: 2,
            sides: [ 2, 2, 2 ]},
          map: Identity,
          logical_routing: DOR{
            order: [ 0, 1, 2 ]},
          opportunistic_hops: true,
          livelock_avoidance: true}],
      legend_name: "DOR-S-Hypercube"}],
  link_classes: [
    LinkClass{ delay: 2 },
    LinkClass{ delay: 2 },
    LinkClass{ delay: 2 },
    LinkClass{ delay: 2 }],
  launch_configurations: [
    Slurm{
      job_pack_size: 1,
      sbatch_args: [ "--exclude=node82,node69,node70,node123" ],
      time: "2-10:00:00"}]}
```