GHG footprint of Slurm job

Quentin D. Read

10/27/2020

Source code for this pdf is available at https://github.com/qdread/energyfootprintslurm.

How much time was saved from code optimization?

The unoptimized code took 2756 seconds (roughly 46 minutes) to run 100 iterations across 8 cores (processors). Assuming that each processor was used for the entire time, that's 2.2048×10^4 processor-seconds, which comes out to 220.48 seconds per iteration. In minutes:

```
unopt_run_time <- 2756
n_iter <- 100
n_cores <- 8

unopt_seconds_run <- unopt_run_time * n_cores
unopt_seconds_per_iter <- unopt_seconds_run / n_iter
unopt_minutes_per_iter <- unopt_seconds_per_iter/60

paste(floor(unopt_minutes_per_iter), 'minutes',
      round((unopt_minutes_per_iter - floor(unopt_minutes_per_iter))*60), 'seconds')
```

```
## [1] "3 minutes 40 seconds"
```

Running 10,000 iterations, that results in approximately 612 processor-hours, which is over 25 processor-days!

The optimized code had an extra setup script that ran once on a single processor for 1066 seconds (roughly 18 minutes). Then, 100 iterations were run in just 29 seconds! That would be 2.32 seconds per iteration, or 2.4266×10^4 processor-seconds (around 7 processor-hours) to run the entire 10,000 iterations, with setup included.

```
opt_setup_time <- 1066
opt_run_time <- 29
n_iter <- 100
n_cores <- 8

opt_seconds_run <- opt_run_time * n_cores
opt_seconds_per_iter <- opt_seconds_run / n_iter

paste(signif(opt_seconds_per_iter, 3), 'seconds')
```

```
## [1] "2.32 seconds"
```

So the amount of time saved is about 606 hours!

```
# Point estimate: time saved
unopt_seconds_per_iter * 10000/3600 - (1066/3600 + opt_seconds_per_iter * 10000/3600)
```

```
## [1] 605.7039
```

GHG intensity of electricity generation in Maryland

We need two data sources from the US Energy Information Administration: the total emissions generated by electricity generation by state, and the total electricity generated by state. Dividing the total emissions by total electricity will give the GHG intensity per unit electricity generated. Despite the fact that EIA is notorious for fancily formatted spreadsheets that are hard to read into R, these spreadsheets are actually reasonably clean.

```r
library(tidyverse)
library(readxl)

clean_names <- function(x) {
  x <- tolower(x)
  x <- gsub("\n", "_", x)
  x <- gsub(" ", "_", x)
  gsub("\\(|\\)", "", x)
}

emission_annual <- read_xls('emission_annual.xls', sheet = 1) %>%
  rename_with(clean_names)
annual_generation_state <-read_xls('annual_generation_state.xls', sheet = 1, skip = 1) %>%
  rename_with(clean_names)
```

What is the most recent year available for Maryland?

```r
emission_annual %>%
  filter(state %in% 'MD', year %in% max(year))
```

```
## # A tibble: 33 x 7
##     year state producer_type energy_source co2_metric_tons so2_metric_tons
##    <dbl> <chr> <chr>         <chr>                   <dbl>           <dbl>
##  1  2019 MD    Electric Uti~ All Sources           1251210               8
##  2  2019 MD    Electric Uti~ Natural Gas           1249309               6
##  3  2019 MD    Electric Uti~ Petroleum                1901               2
##  4  2019 MD    IPP NAICS-22~ All Sources           9098812            4141
##  5  2019 MD    IPP NAICS-22~ Coal                  5084044            4009
##  6  2019 MD    IPP NAICS-22~ Natural Gas           3686163              17
##  7  2019 MD    IPP NAICS-22~ Other Biomass               0               0
##  8  2019 MD    IPP NAICS-22~ Other                  257125              19
##  9  2019 MD    IPP NAICS-22~ Petroleum               71480              96
## 10  2019 MD    IPP NAICS-22~ All Sources           1858966            1009
## # ... with 23 more rows, and 1 more variable: nox_metric_tons <dbl>
```

```r
annual_generation_state %>%
  filter(state %in% 'MD', year %in% max(year))
```

```
## # A tibble: 44 x 5
##     year state type_of_producer        energy_source        generation_megawat~
##    <dbl> <chr> <chr>                   <chr>                              <dbl>
##  1  2019 MD    Total Electric Power I~ Total                           39328689
##  2  2019 MD    Total Electric Power I~ Coal                             5721573
##  3  2019 MD    Total Electric Power I~ Hydroelectric Conven~            2188051
##  4  2019 MD    Total Electric Power I~ Natural Gas                     14605261
```

```
## 5 2019 MD    Total Electric Power I~ Nuclear                          15012922
## 6 2019 MD    Total Electric Power I~ Other                              328530
## 7 2019 MD    Total Electric Power I~ Petroleum                           67269
## 8 2019 MD    Total Electric Power I~ Solar Thermal and Ph~              494311
## 9 2019 MD    Total Electric Power I~ Other Biomass                      330379
## 10 2019 MD   Total Electric Power I~ Wind                               520269
## # ... with 34 more rows
```

Let's just use the "Total Electric Power Industry" rows because I'm not sure how the subcategories match up between the two.

```
md_emission <- emission_annual %>%
  filter(state %in% 'MD', year %in% max(year),
         producer_type %in% 'Total Electric Power Industry')

md_generation <- annual_generation_state %>%
  filter(state %in% 'MD', year %in% max(year),
         type_of_producer %in% 'Total Electric Power Industry')
```

Since we don't know the mix of sources used to power Park Place versus UMD, we will have to just stick with the total. We can ignore the SO2 and NOx columns. We have tonnes per MWh, which is equivalent to kg per kWh.

```
total_emissions <- with(md_emission,
                        co2_metric_tons[energy_source %in% 'All Sources'])
total_generation <- with(md_generation,
                         generation_megawatthours[energy_source %in% 'Total'])

total_emissions / total_generation
```

```
## [1] 0.3333197
```

So about 333 grams of CO2 are released to provide 1 kWh of electricity in Maryland, averaging across all possible modes.

How much electricity does it take to run the Slurm cluster?

We found a number of different sources for this. It seems like the characteristics of the hardware have a pretty big impact on how much power it draws, which makes sense. How computationally intensive the job is doesn't have as big of an impact. Based on the different sources, it looks like a single Slurm processor core draws somewhere between 15 and 50 W when running a job (based on estimates of 60-200 W for a 4-core processor). I'm assuming a triangular distribution between 15 and 50, with a peak in the middle at 33 W. (for now)

What's the CO2 footprint of the two jobs? And the difference between them?

That is easily calculated from the unoptimized and optimized run times, and the range of values for power consumption.

```
unopt_time <- unopt_seconds_per_iter * 10000
unopt_energy <- unopt_time/3600 * c(15, 32.5, 50)/1000 # in kWh
unopt_co2 <- unopt_energy * total_emissions / total_generation

opt_time <- 1066 + opt_seconds_per_iter * 10000
opt_energy <- opt_time/3600 * c(15, 32.5, 50)/1000 # in kWh
```

```r
opt_co2 <- opt_energy * total_emissions / total_generation

# Estimate for kg CO2 from unoptimized code
unopt_co2
```

```
## [1]  3.062097  6.634543 10.206989
```

```r
# Estimate for kg CO2 from optimized code
opt_co2
```

```
## [1] 0.03370140 0.07301969 0.11233799
```

```r
# Point estimate for kg CO2 saved.
unopt_co2[2] - opt_co2[2]
```

```
## [1] 6.561523
```

Our point estimate is about 6.56 kg CO2 saved!

Put that in terms I can understand

Let's compare the CO2 footprint of the energy saved with the CO2 footprint of some other activities:

- Driving a passenger car
- Streaming Netflix
- Eating a hamburger

For the car, we can use EPA's number of 404 g/mile (for now). For Netflix, it's a bit more uncertain since there are a number of sources of emissions: the data center, the data transmission, and the device you are streaming on. The CO2 footprint of Wifi versus 4G is pretty different (much higher for 4G). Obviously larger-screen devices have a bigger footprint. Let's use the weighted average of 70 g per hour across all modes of data transmission and whatever device you might use.

Assuming EPA's number for GHG intensity of driving a passenger car, the amount of CO2 saved would get us 16.2 miles or 26.1 kilometers in a car! That would get you from Annapolis across the Bay Bridge and to the far side of Kent Island.

Using the weighted average across devices, with the CO2 saved from optimization, you would be able to kick back and stream Netflix for 93.7 hours, or about 4 days straight! That's almost exactly enough to watch every episode of "Great British Baking Show" ever recorded.

You'd only be able to produce 1.6 Big Macs with that amount of CO2, though, which is probably a bad thing for several reasons.

```r
# Point estimate for km driven
units::set_units((unopt_co2[2] - opt_co2[2])/0.404, 'mi') %>% units::set_units('km')
```

```
## 26.13799 [km]
```

```r
# Point estimate for hours streaming
(unopt_co2[2] - opt_co2[2])/0.07
```

```
## [1] 93.73605
```

```r
# Point estimate for number of Big macs
(unopt_co2[2] - opt_co2[2])/4
```

```
## [1] 1.640381
```

Data sources for processor energy use

Basmadjian, R. & de Meer, H. (2012). Evaluating and modeling power consumption of multi-core processors. In: Proceedings of the 3rd International Conference on Future Energy Systems Where Energy, Computing and Communication Meet - e-Energy '12. Presented at the the 3rd International Conference, ACM Press, Madrid, Spain, pp. 1–10.

Jarus, M., Oleksiak, A., Piontek, T. & Węglarz, J. (2014). Runtime power usage estimation of HPC servers for various classes of real-life applications. Future Generation Computer Systems, Special Section: Intelligent Big Data Processing, 36, 299–310.

Mantovani, F. & Calore, E. (2018). Performance and Power Analysis of HPC Workloads on Heterogenous Multi-Node Clusters. JLPEA, 8, 13.