# Tool for configuration of coordinated cache and bank coloring

Bjorn Andersson

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.



DM21-0235

**Background:**

*** About multicore.** Multicore processors are the norm today. This has been the case in desktop and laptop computers for 15 years. This has been the case for mobile computing for 10 years. In DoD, for non-safety-critical systems, multicore system have started to be used. In DoD, for safety-critical systems, multicores (with all cores enabled) are currently not used but there is great interest in finding ways to do so.

*** About multicore and timing.** Multicore processors tend to have shared memory (both DRAM memory shared between processor cores and shared physical address space between processor cores). There are also many resources that aim to speed up execution (e.g., cache) and there are resources for connecting processor cores to memory (e.g., memory bus). On a higher level, one can say that there are resources in the memory system that are shared between processor cores and hence they may be shared between processes/threads on different processor cores. For some of these resources, it holds that the resource can only be used by one thread/process at a time (e.g., memory bus). For other resources, it holds that the use of one resource by one thread causes another thread later to execute more slowly (e.g., cache eviction).

* **Mechanism.** The SEI has developed a technique called coordinated cache and bank coloring. See N. Suzuki et al., "Coordinated bank and cache coloring for temporal protection of memory accesses," ICESS 2013.

This technique involves changing the virtual memory manager in the operating system. It also involves deciding how the virtual memory manager should work; we call it configuration. This slide deck presents the tool for configuration.

* **The advantages of the mechanism.** For a mechanism, two properties are desirable:
    1. Providing timing isolation with respect to both cache and bank.
    2. Requiring only the use of the virtual memory system (i.e., not depend on instructions or features that are available in only few processors). This is important for systems that may have long life-time (decades).

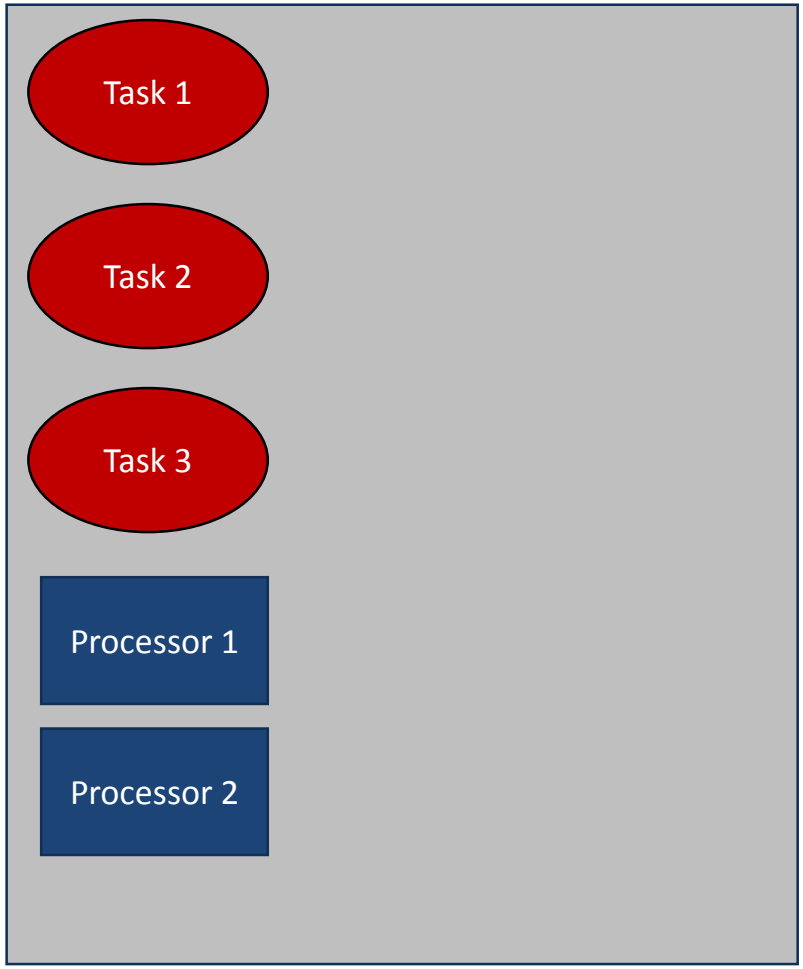
  As far as we know, coordinated cache and bank coloring is the only technique that offers these two properties. (there are other techniques for cache coloring but they do not achieve bank coloring; there are techniques for bank coloring but they do not achieve cache coloring; there are techniques that use cache locking but this requires special instructions).

* **The disadvantages of the mechanism:**
    1. Since it requires changing the behavior of the virtual memory system, it requires changing the operating system.
    2. May require that at most a fraction of the available physical memory is used.

  There are ways to alleviate these drawbacks. For example, for the former, it is sometimes possible to use kernel modules. For the latter, one can distinguish hard and soft real-time and take advantage of that.

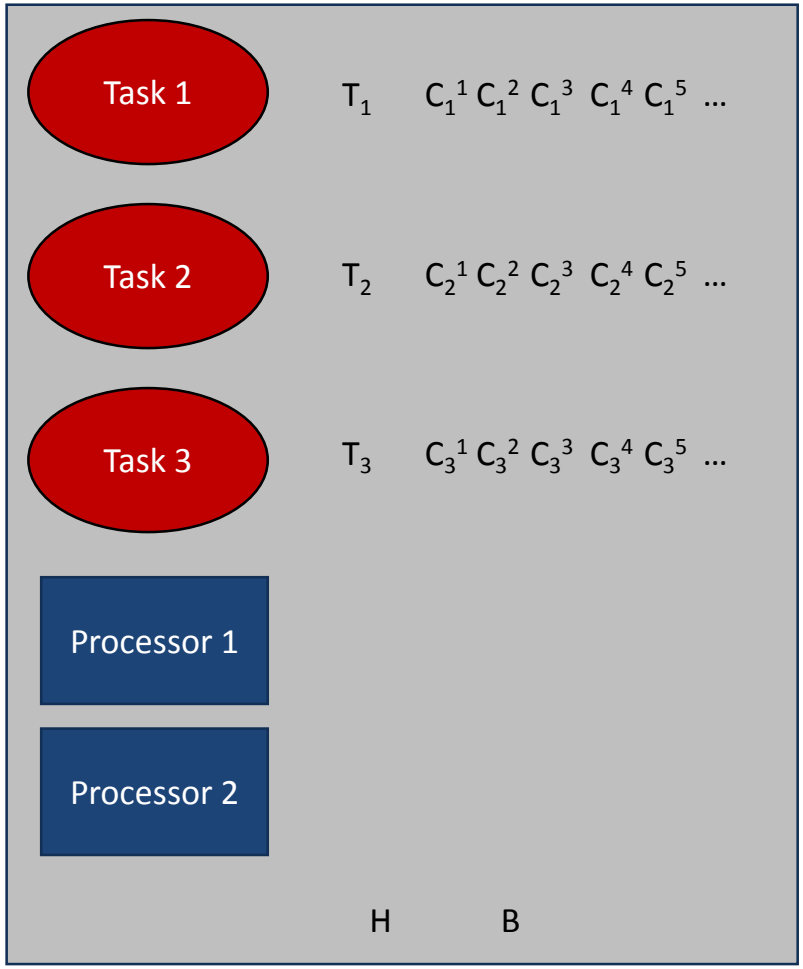
# What the configuration tool does

Model of the system



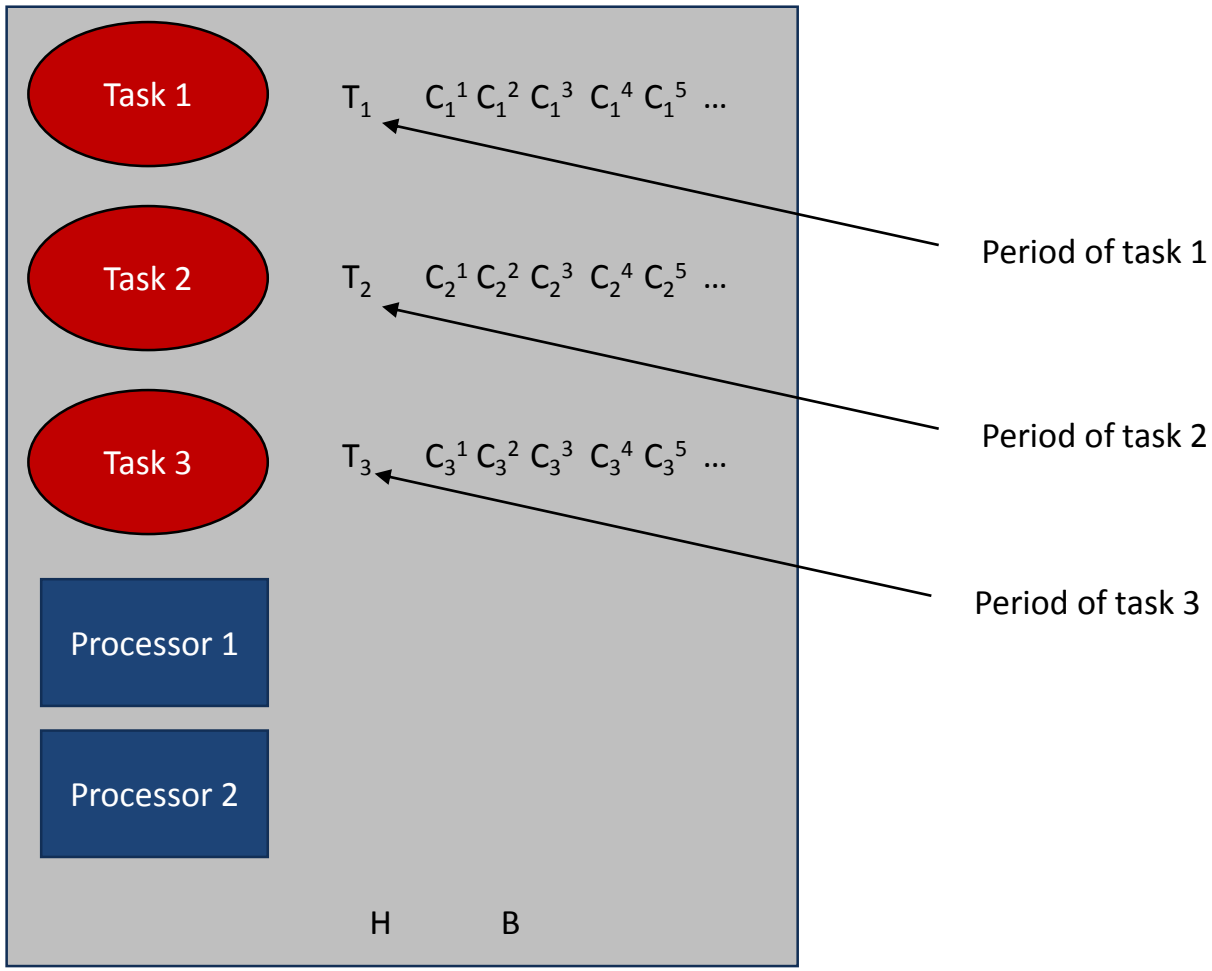
# What the configuration tool does

Model of the system



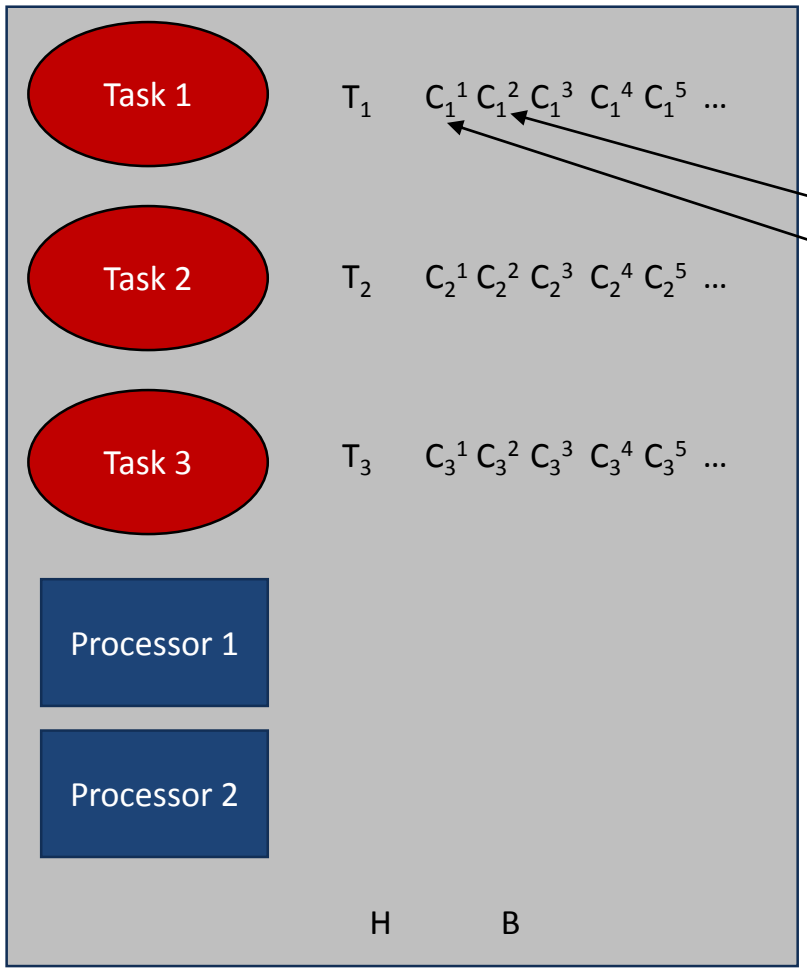
# What the configuration tool does

Model of the system



# What the configuration tool does

Model of the system

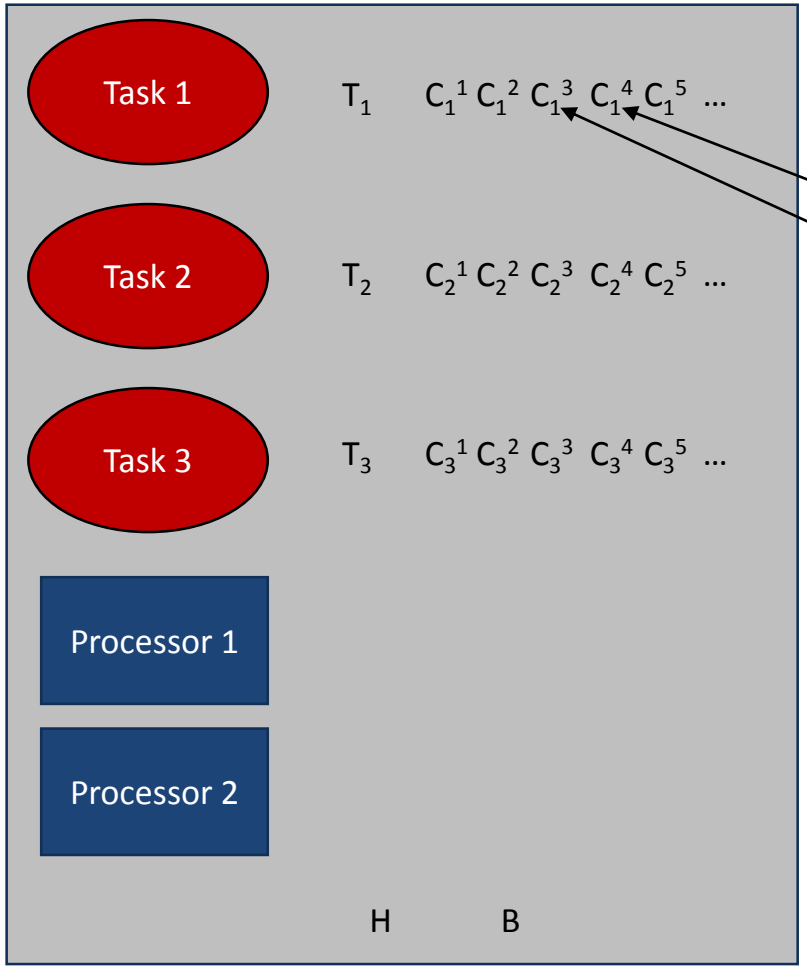



Execution time of task 1 if it is assigned 2 cache colors

Execution time of task 1 if it is assigned 1 cache color

# What the configuration tool does

Model of the system

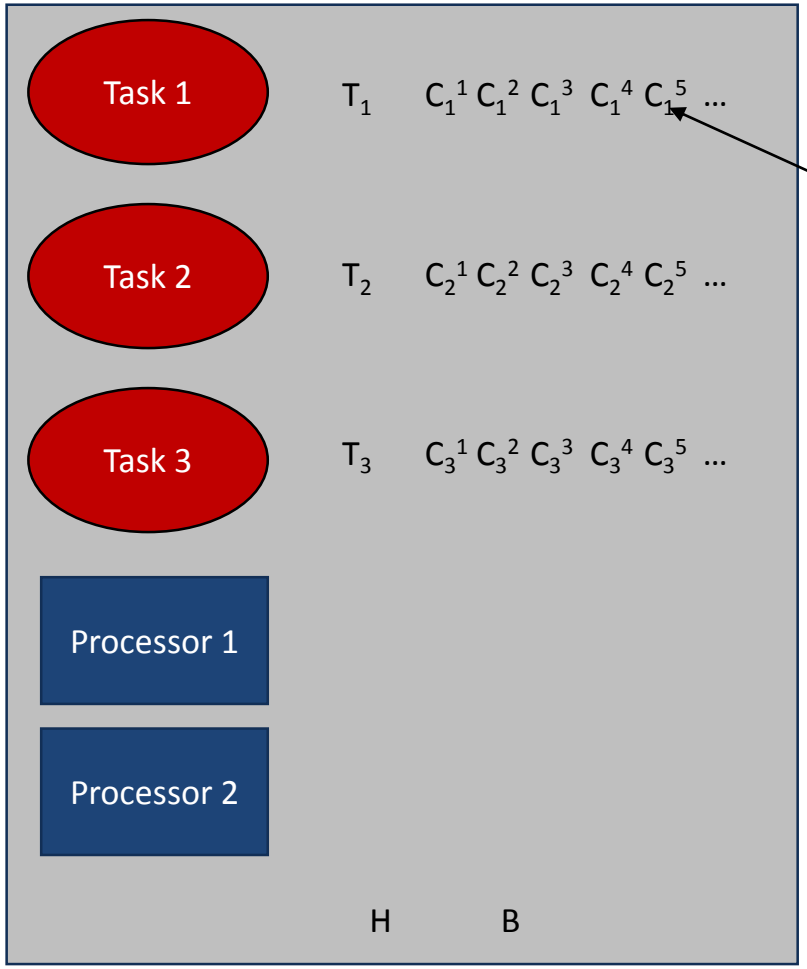



Execution time of task 1 if it is assigned 4 cache colors

Execution time of task 1 if it is assigned 3 cache colors

# What the configuration tool does

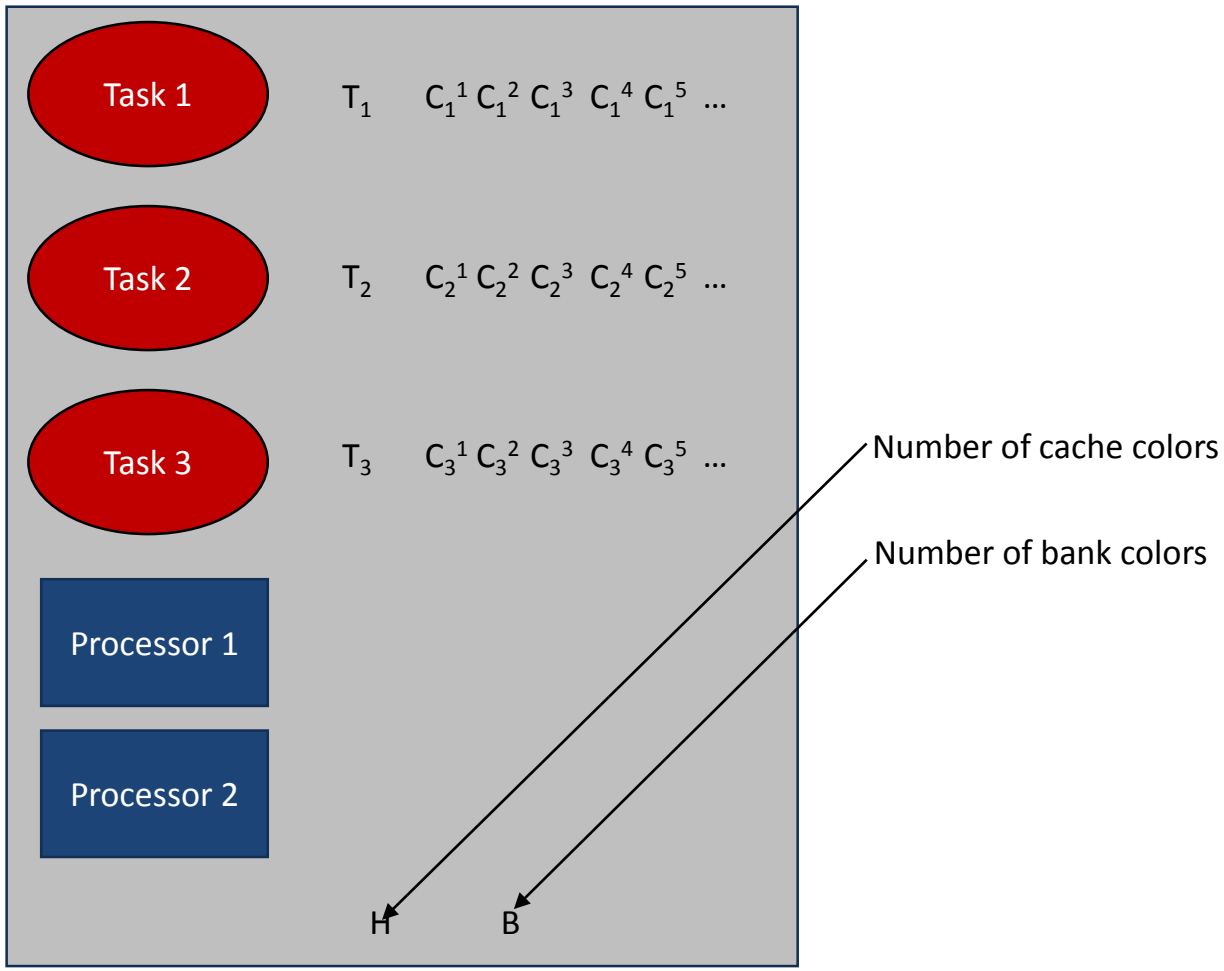
Model of the system



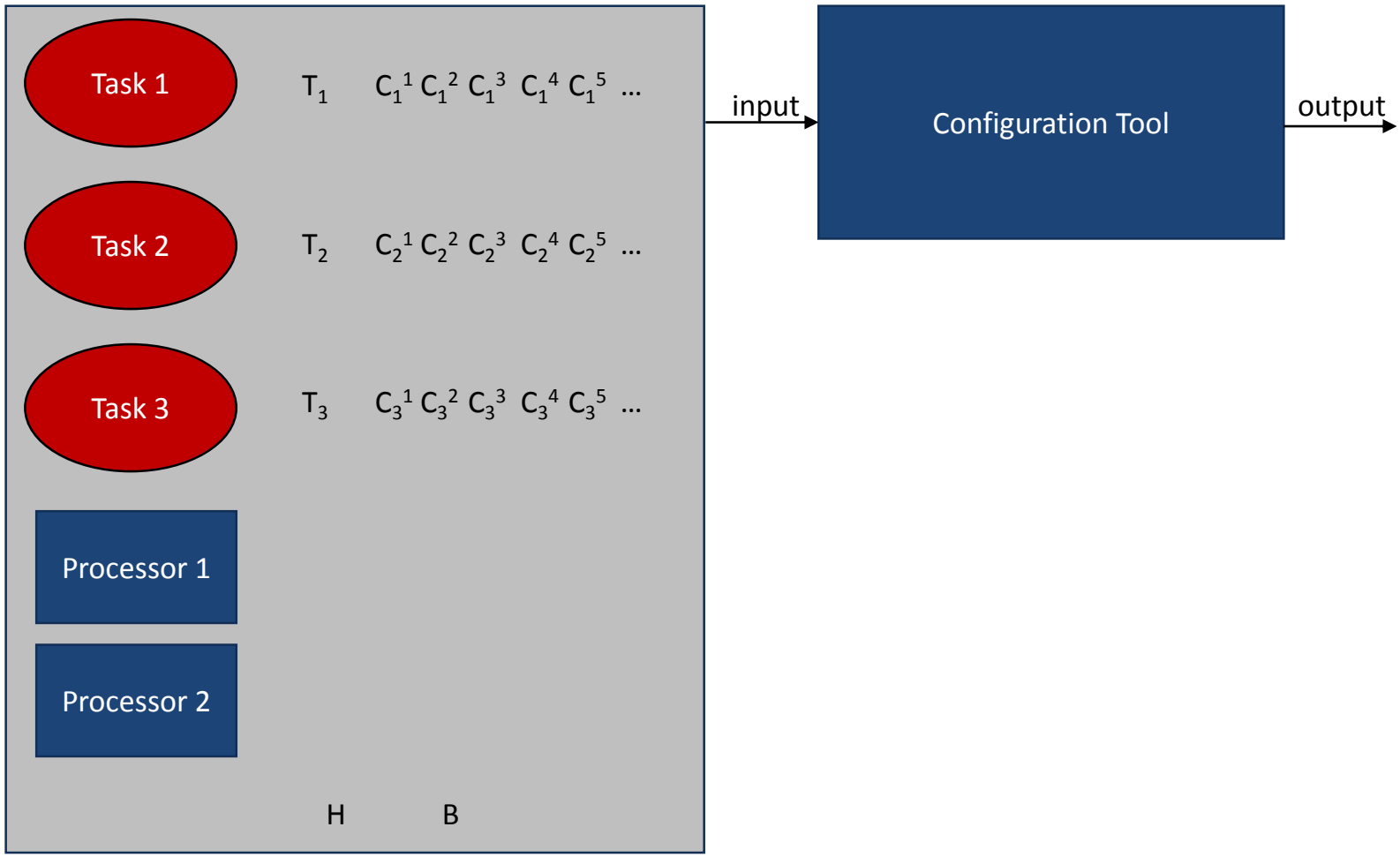

Execution time of task 1 if it is assigned 5 cache colors

# What the configuration tool does

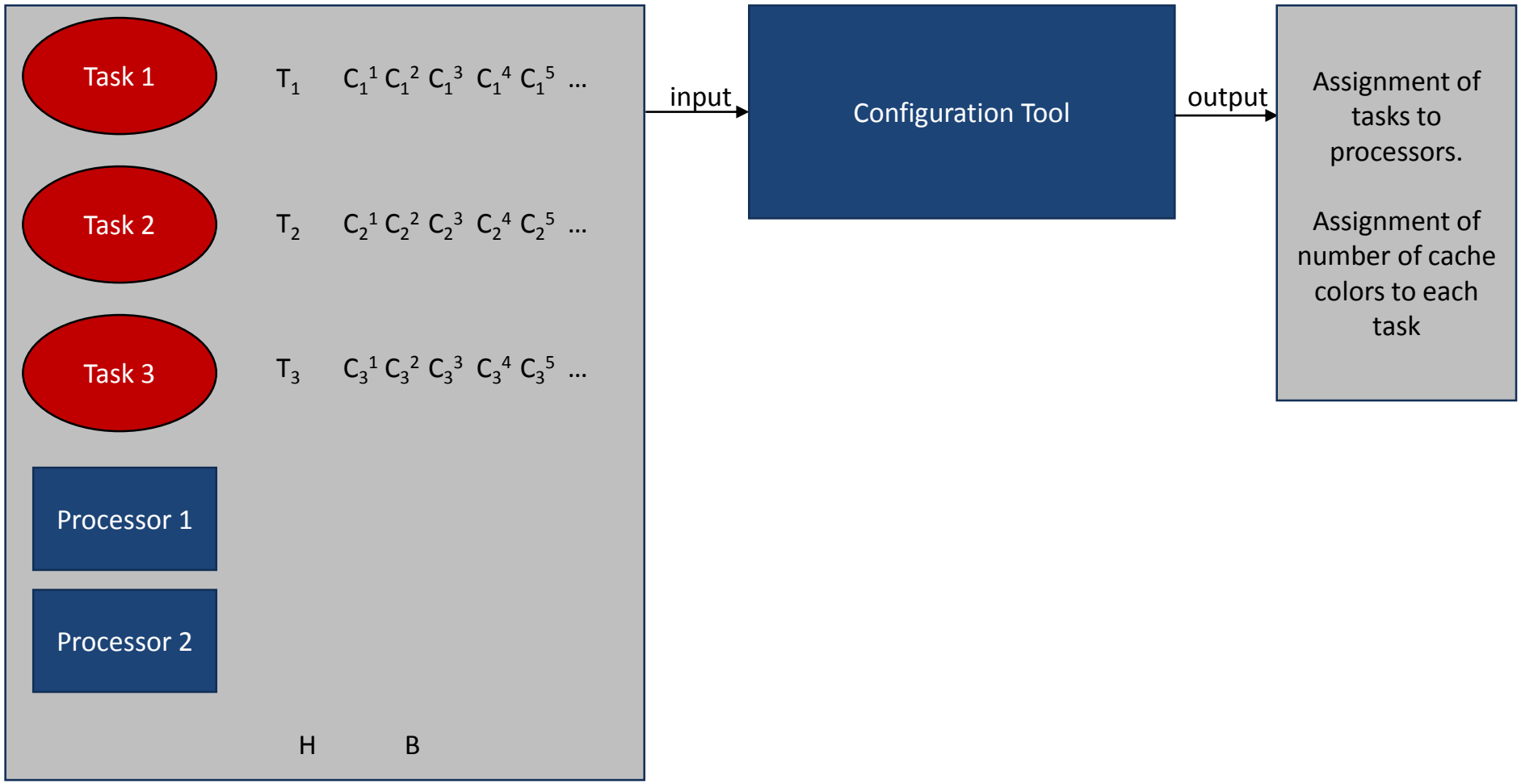
Model of the system



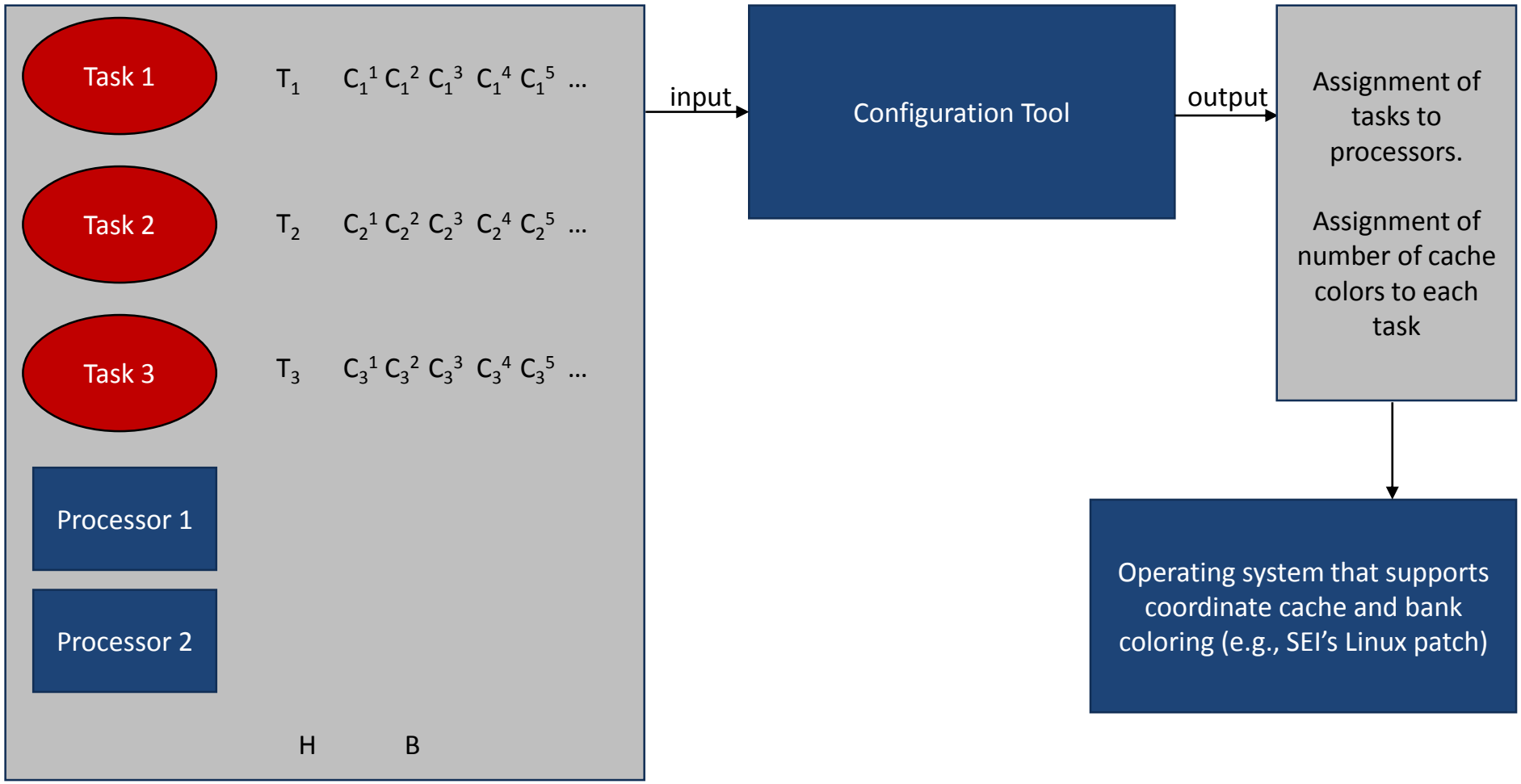
# What the configuration tool does



# What the configuration tool does

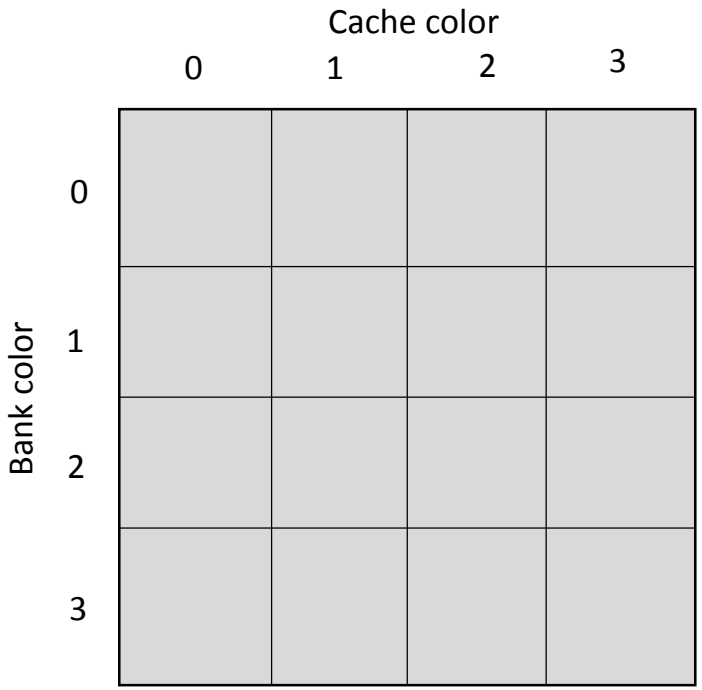


# What the configuration tool does



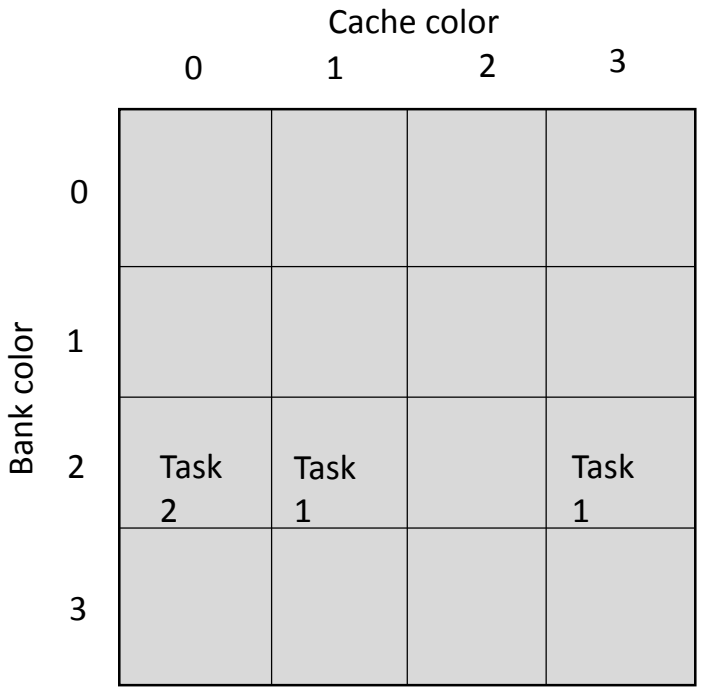
# Understanding the constraints on configuration

Consider a computer system with 4 cache colors and 4 bank colors and two tasks $\tau_1$ and $\tau_2$.
The physical memory consists of 16 cells.
Assume task 1 and task 2 are assigned to the same processor, processor 1.



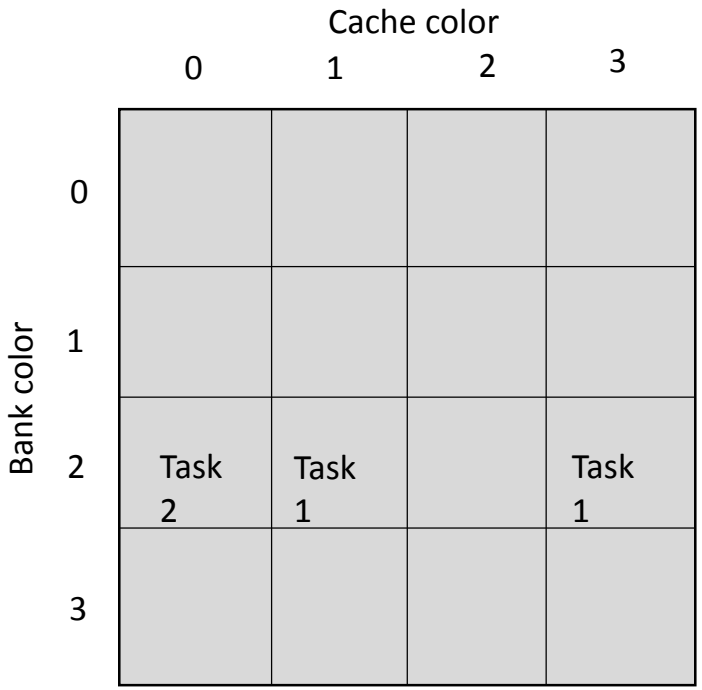
# Understanding the constraints on configuration

Consider a computer system with 4 cache colors and 4 bank colors and two tasks $\tau_1$ and $\tau_2$.
The physical memory consists of 16 cells.
Assume task 1 and task 2 are assigned to the same processor, processor 1.



# Understanding the constraints on configuration

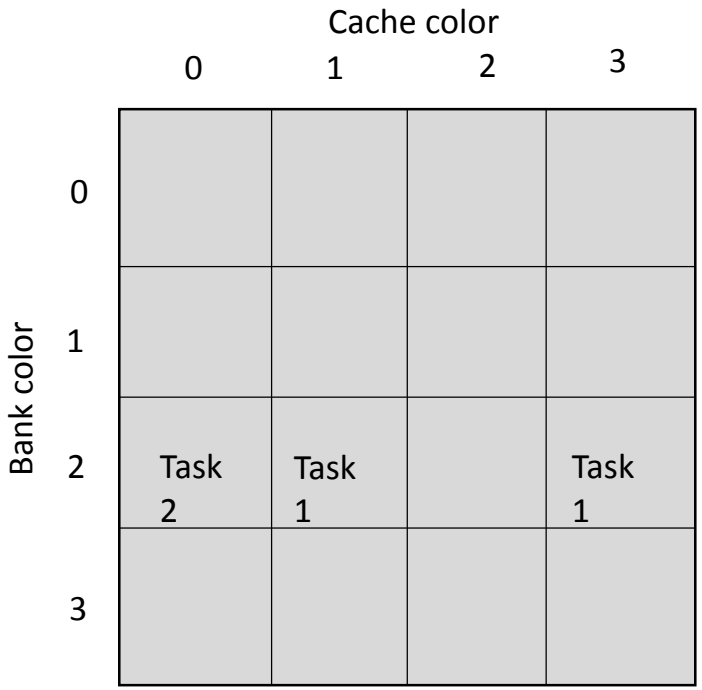
Consider a computer system with 4 cache colors and 4 bank colors and two tasks $\tau_1$ and $\tau_2$.
The physical memory consists of 16 cells.
Assume task 1 and task 2 are assigned to the same processor, processor 1.




Note that task 1 and task 2 do not share cache colors.

# Understanding the constraints on configuration

Consider a computer system with 4 cache colors and 4 bank colors and two tasks $\tau_1$ and $\tau_2$.
The physical memory consists of 16 cells.
Assume task 1 and task 2 are assigned to the same processor, processor 1.

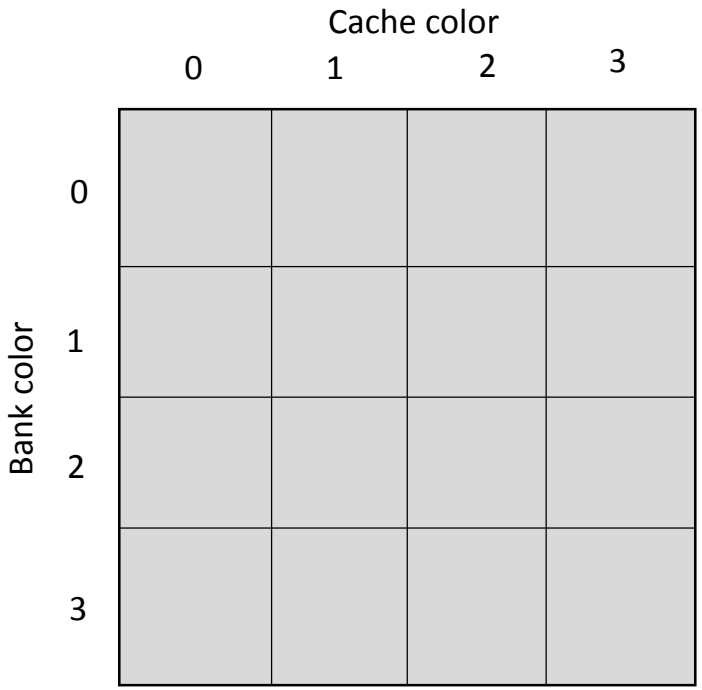



Note that task 1 and task 2 shares bank colors.

But this is OK because they are assigned to the same processor.

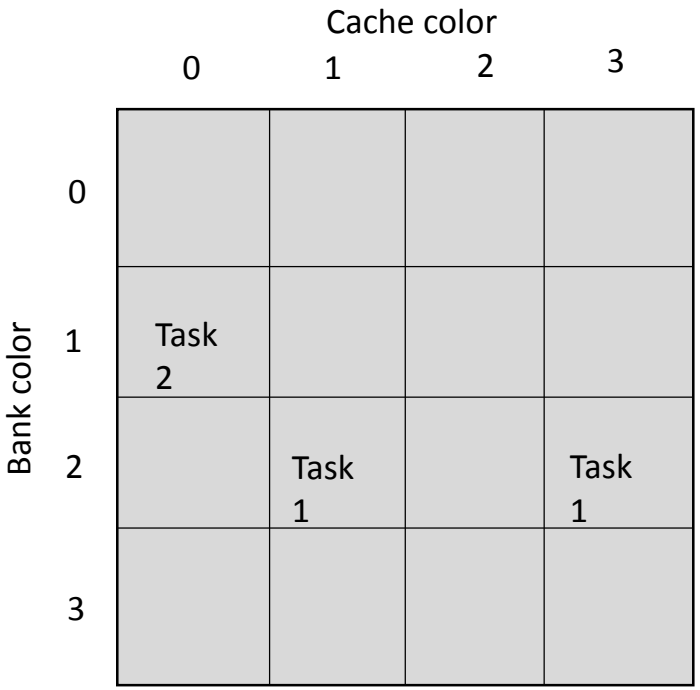
# Understanding the constraints on configuration

Consider a computer system with 4 cache colors and 4 bank colors and two tasks $\tau_1$ and $\tau_2$.
The physical memory consists of 16 cells.
Assume task 1 and task 2 are assigned to different processors, processor 1 and processor 2 respectively.



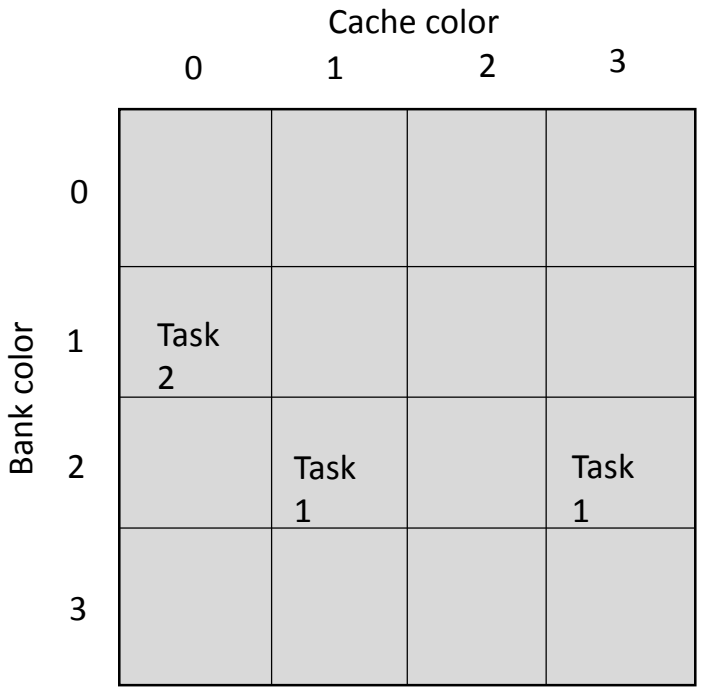
# Understanding the constraints on configuration

Consider a computer system with 4 cache colors and 4 bank colors and two tasks $\tau_1$ and $\tau_2$.
The physical memory consists of 16 cells.
Assume task 1 and task 2 are assigned to different processors, processor 1 and processor 2 respectively.



# Understanding the constraints on configuration

Consider a computer system with 4 cache colors and 4 bank colors and two tasks $\tau_1$ and $\tau_2$.
The physical memory consists of 16 cells.
Assume task 1 and task 2 are assigned to different processors, processor 1 and processor 2 respectively.

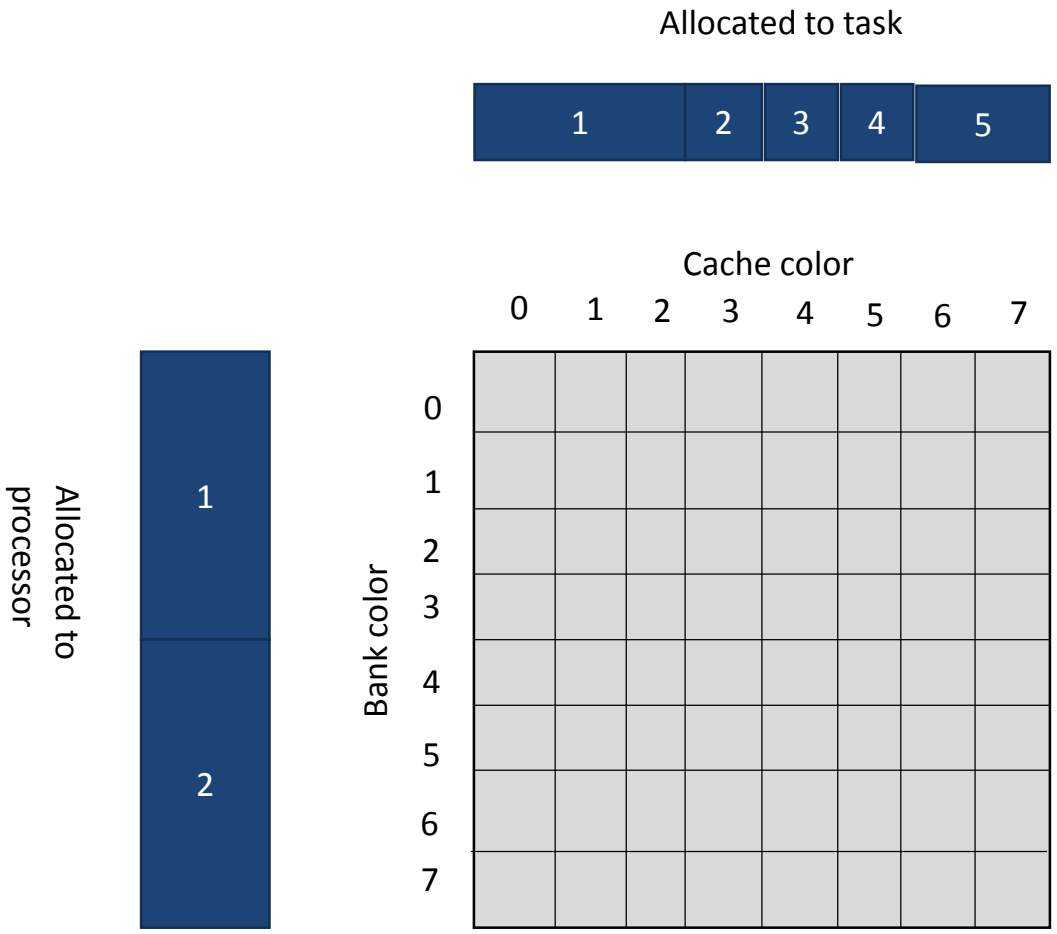



Note that task 1 and task 2 are assigned to different cache colors and bank colors.

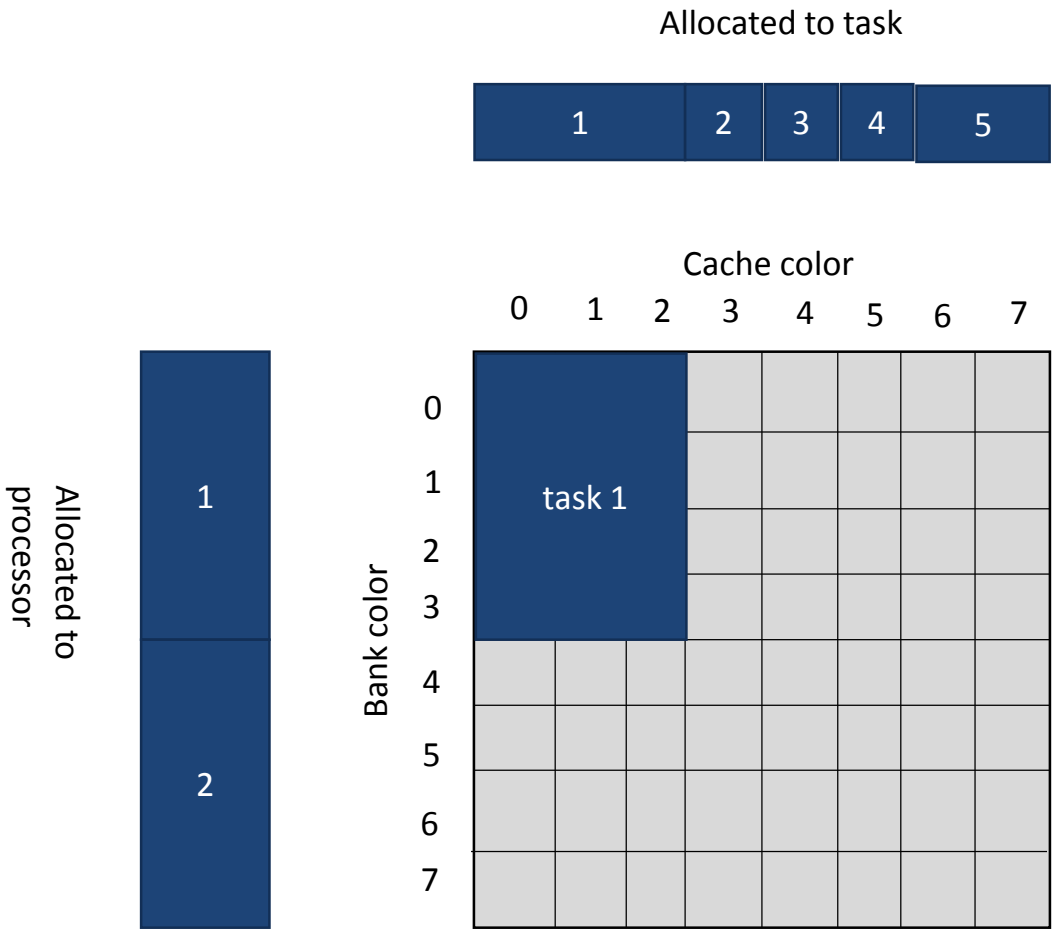
Since task 1 and task 2 are assigned to different processors, they must use different bank colors.

The figure on previous slide shows a concrete allocation. We will however create an abstract allocation which can then be converted to a concrete allocation. The next slide shows an abstract allocation.

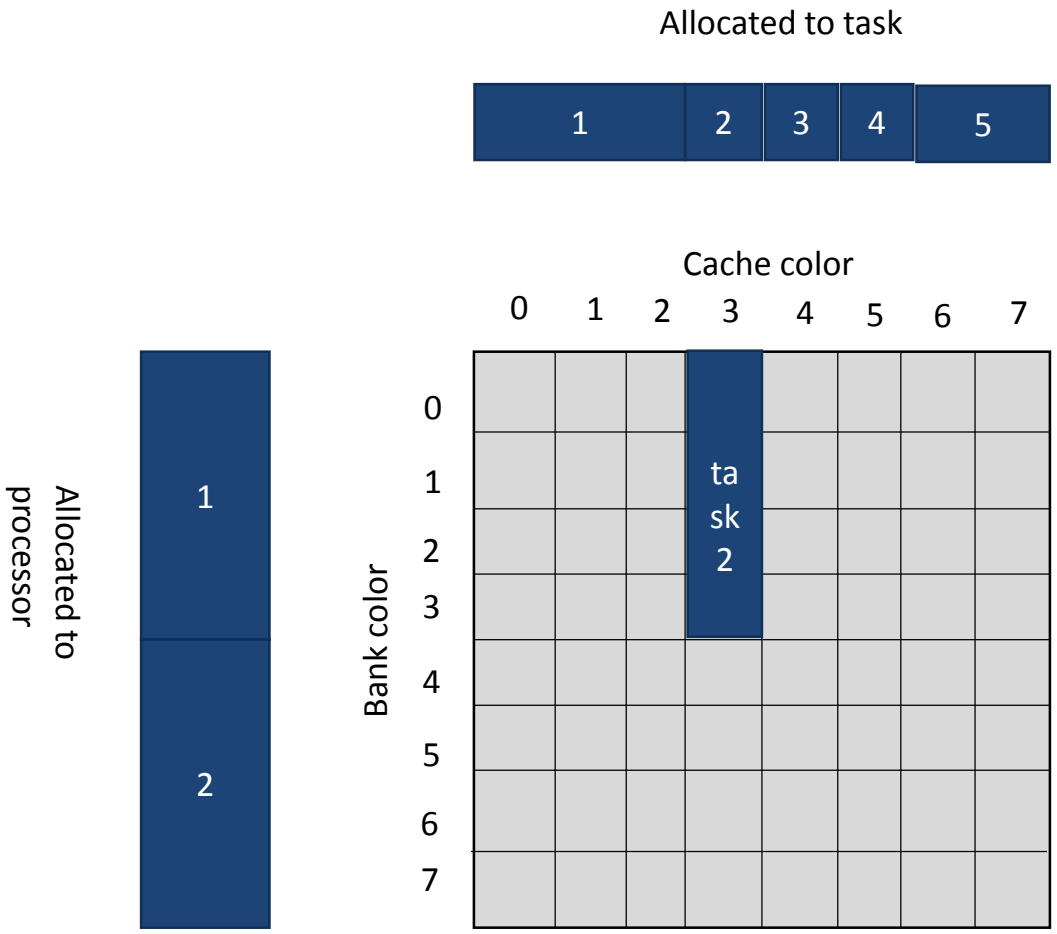
Assume task 1 and task 2 are assigned to processor 1 and task 3, task 4, and task 5 are assigned to processor 2.



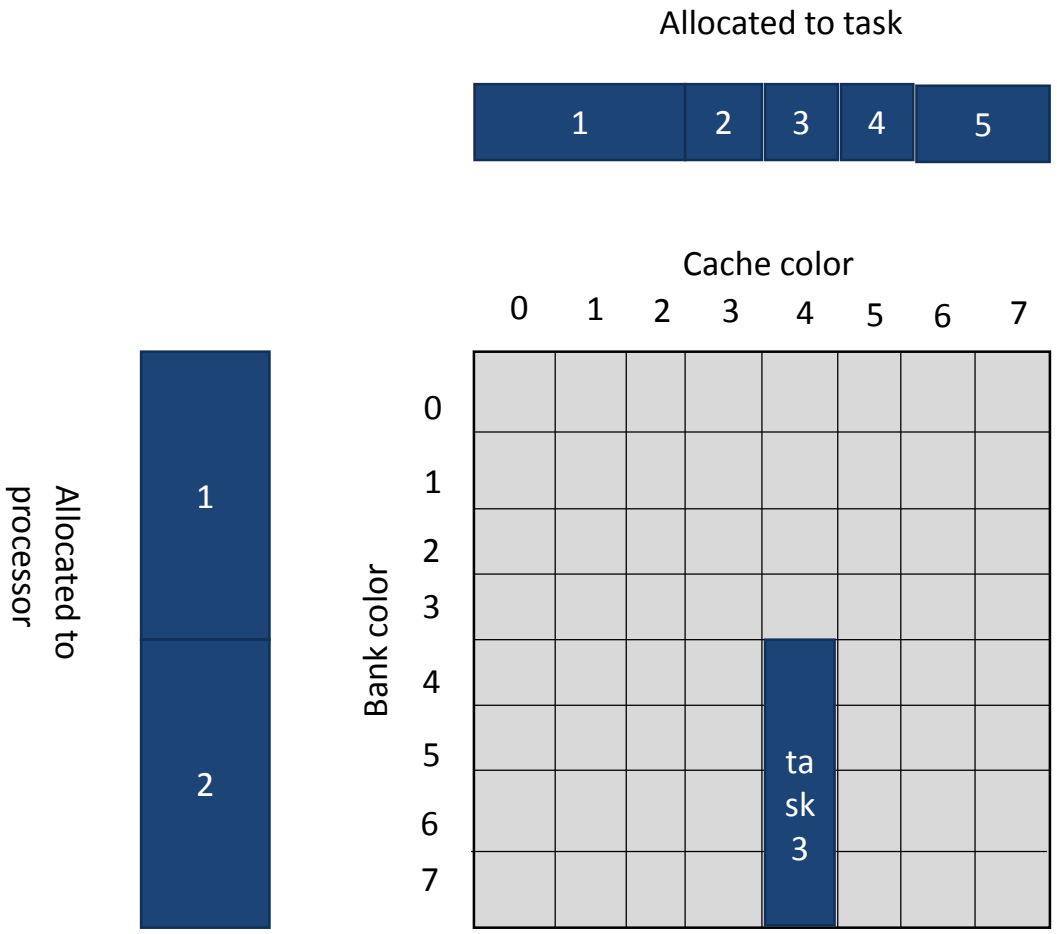
Assume task 1 and task 2 are assigned to processor 1 and task 3, task 4, and task 5 are assigned to processor 2.



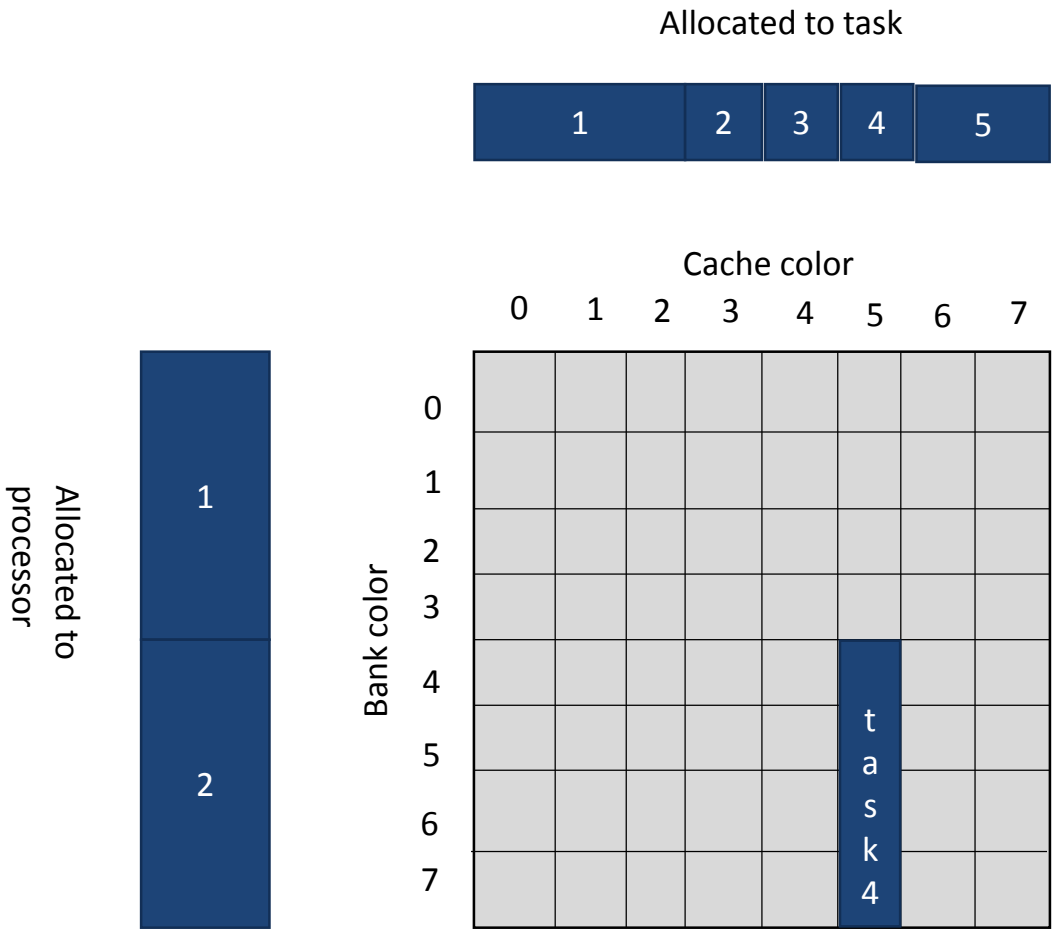
Assume task 1 and task 2 are assigned to processor 1 and task 3, task 4, and task 5 are assigned to processor 2.



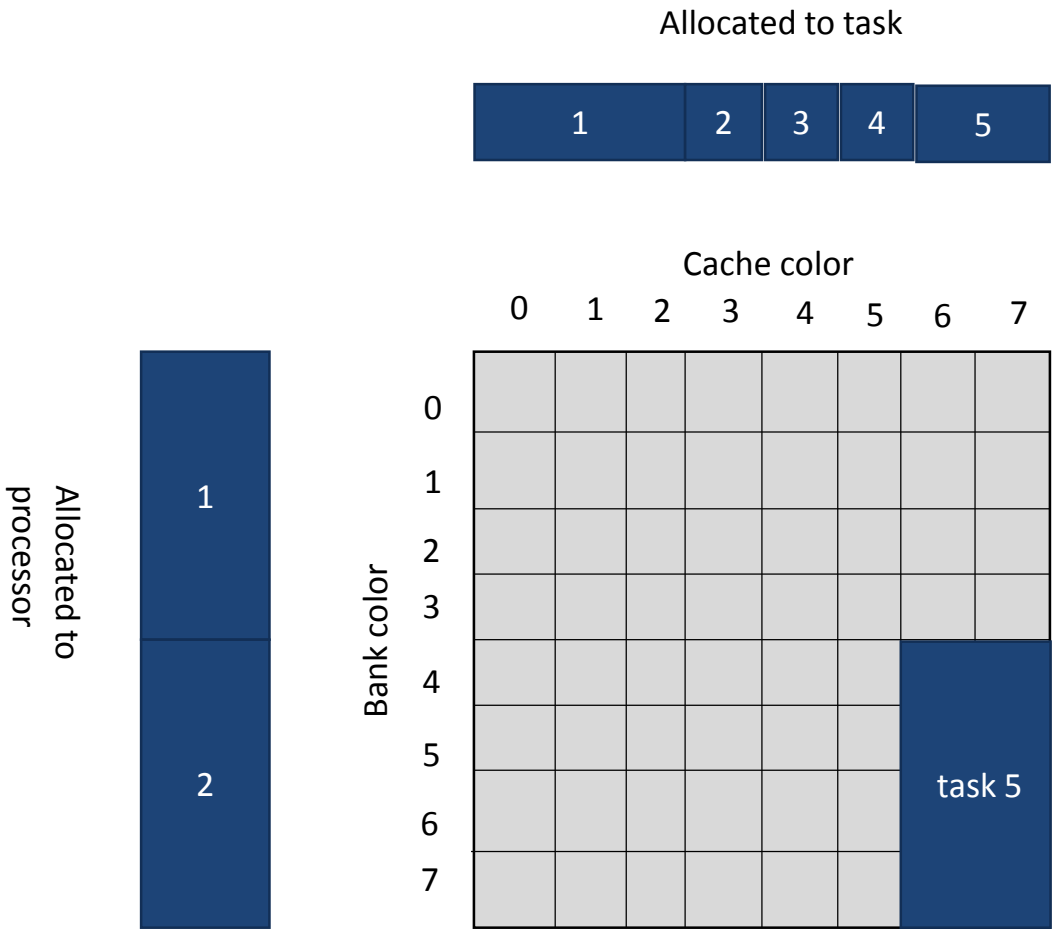
Assume task 1 and task 2 are assigned to processor 1 and task 3, task 4, and task 5 are assigned to processor 2.



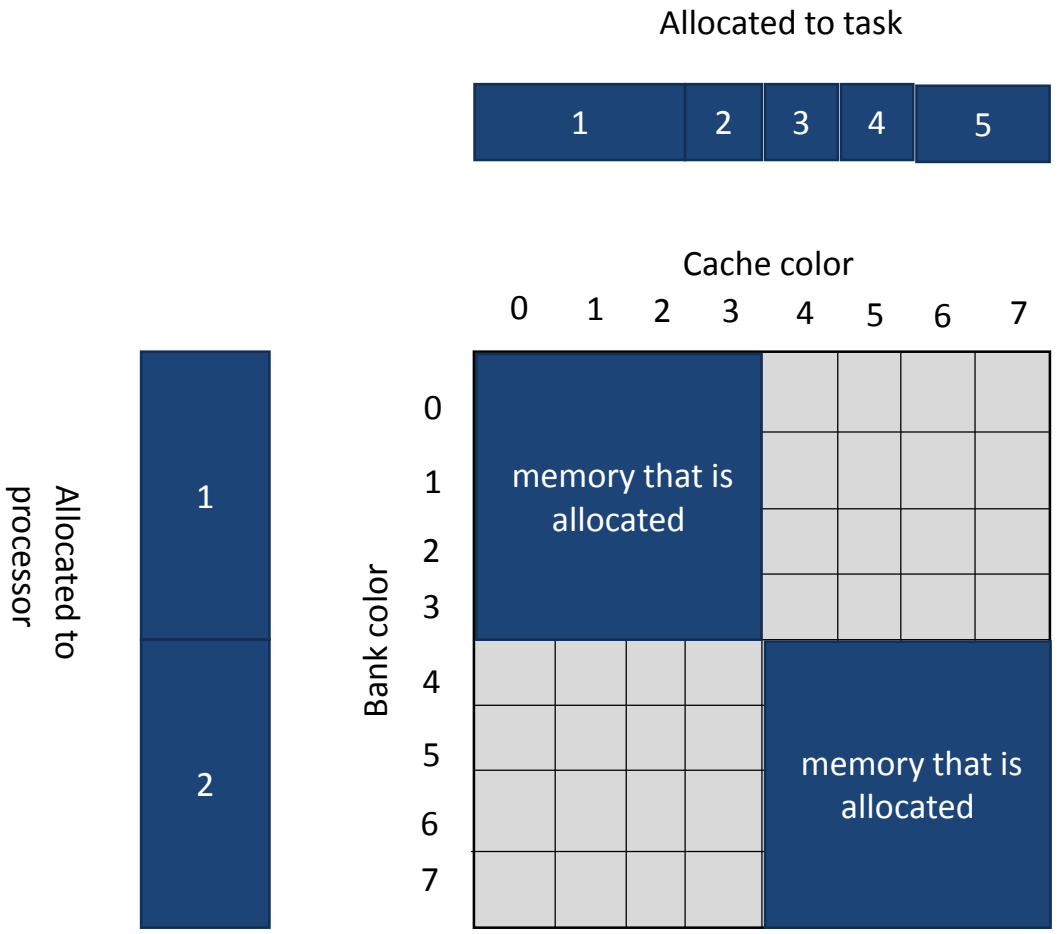
Assume task 1 and task 2 are assigned to processor 1 and task 3, task 4, and task 5 are assigned to processor 2.



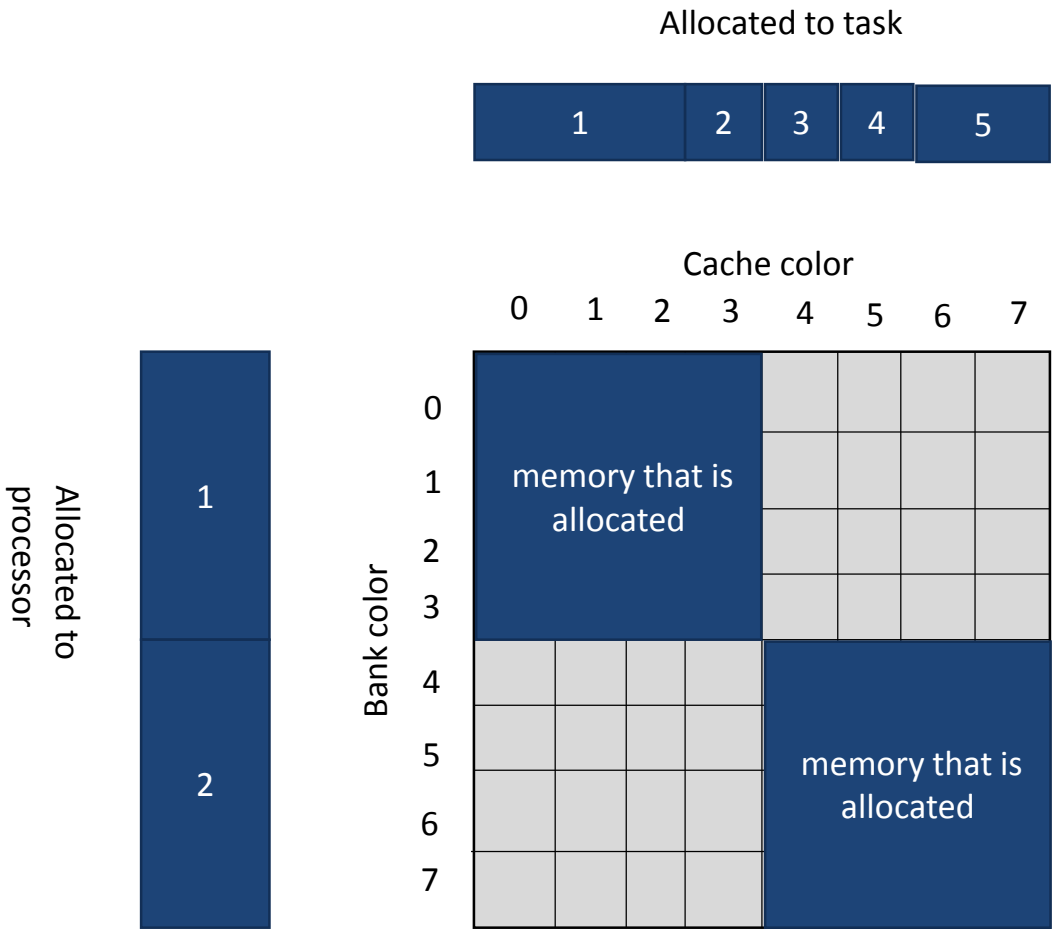
Assume task 1 and task 2 are assigned to processor 1 and task 3, task 4, and task 5 are assigned to processor 2.



Assume task 1 and task 2 are assigned to processor 1 and task 3, task 4, and task 5 are assigned to processor 2.



Assume task 1 and task 2 are assigned to processor 1 and task 3, task 4, and task 5 are assigned to processor 2.

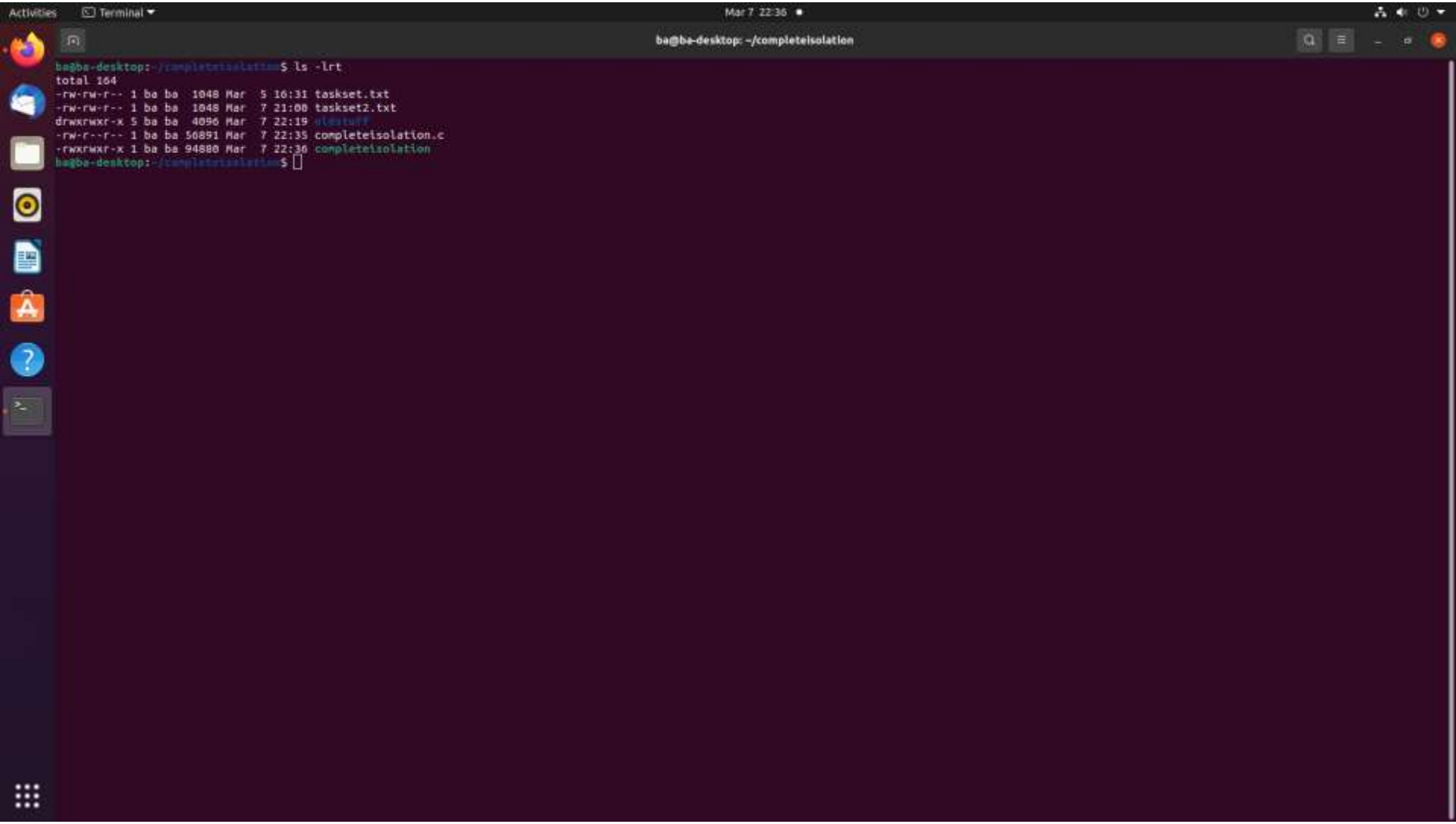


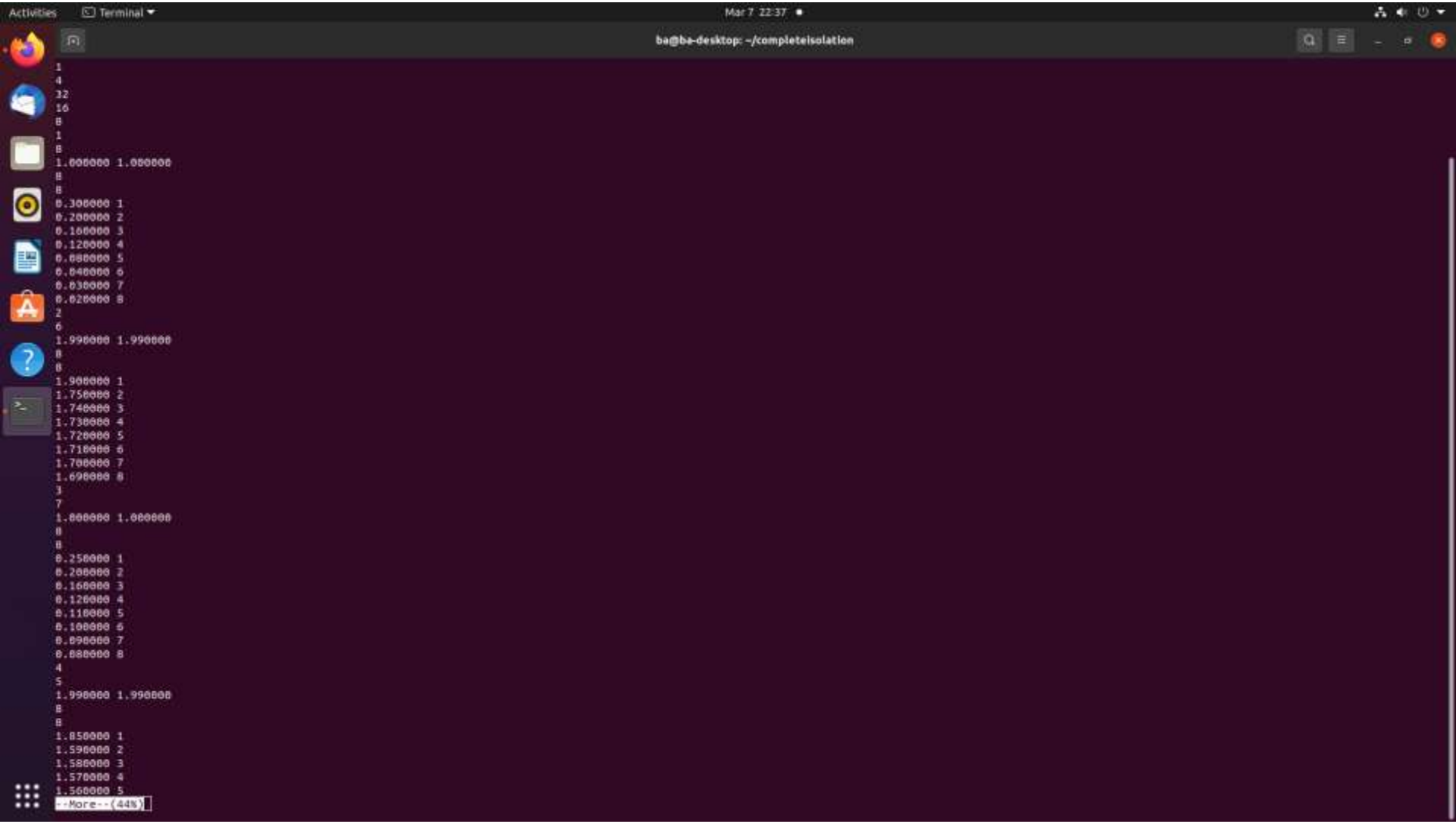

Our coordinated cache and bank coloring forces us to waste 50% of the memory on a dual core system.

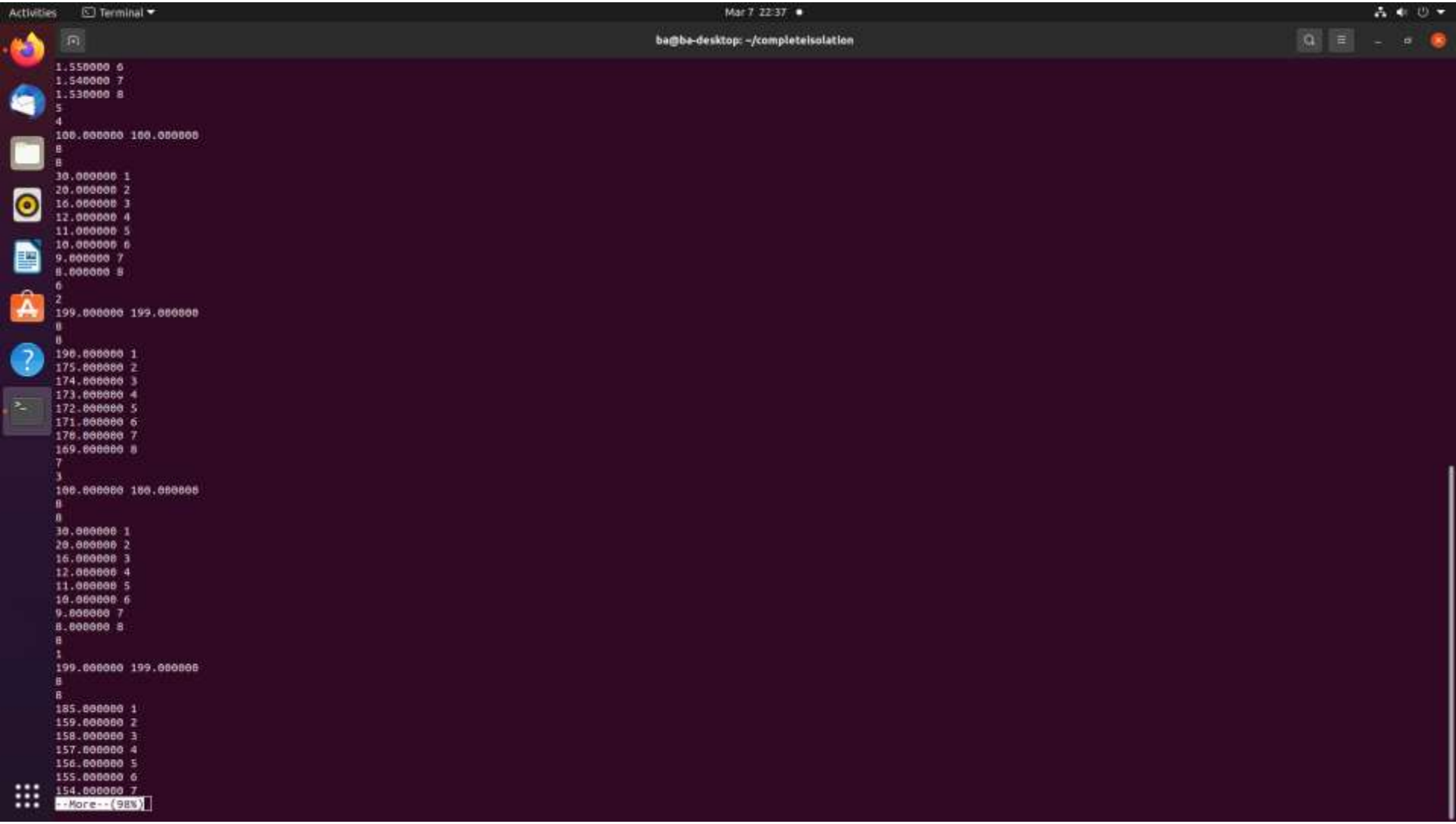
The configuration tool can be downloaded at

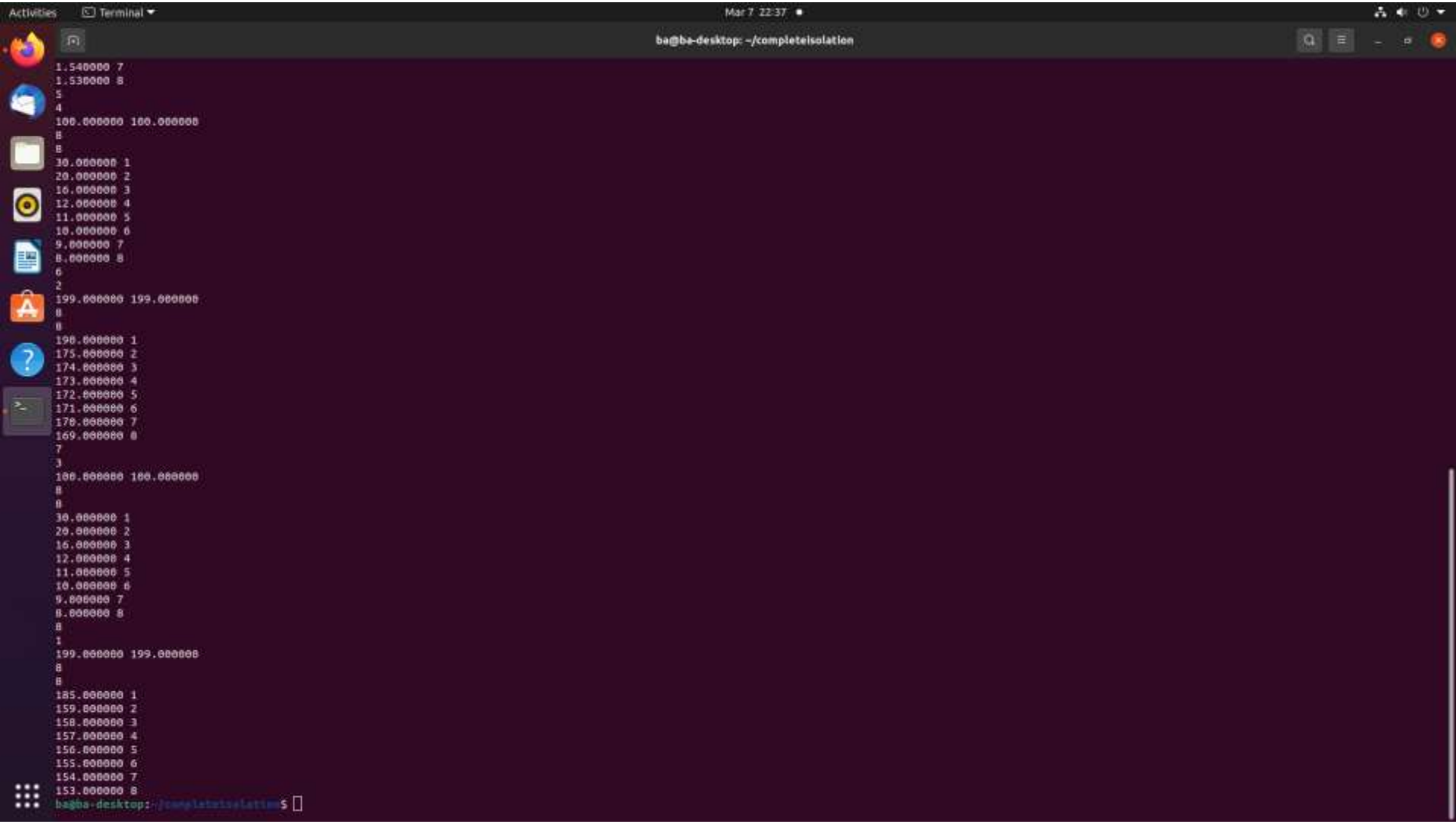
http://www.andrew.cmu.edu/user/banderss/software/coordinated_cache_and_bank_coloring_configuration/completeisolation.c.

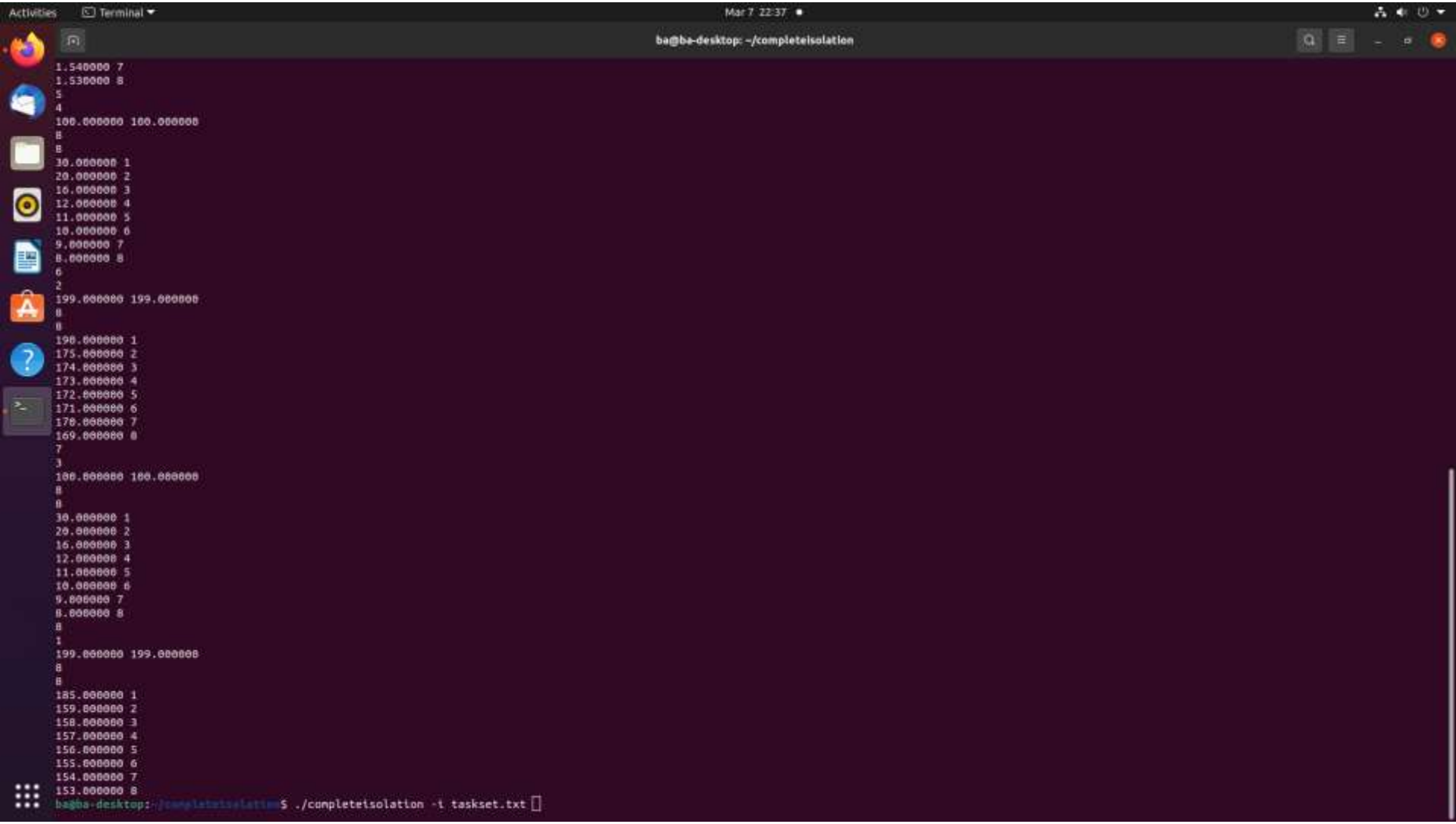
Let us now see some screenshot from the use of the tool

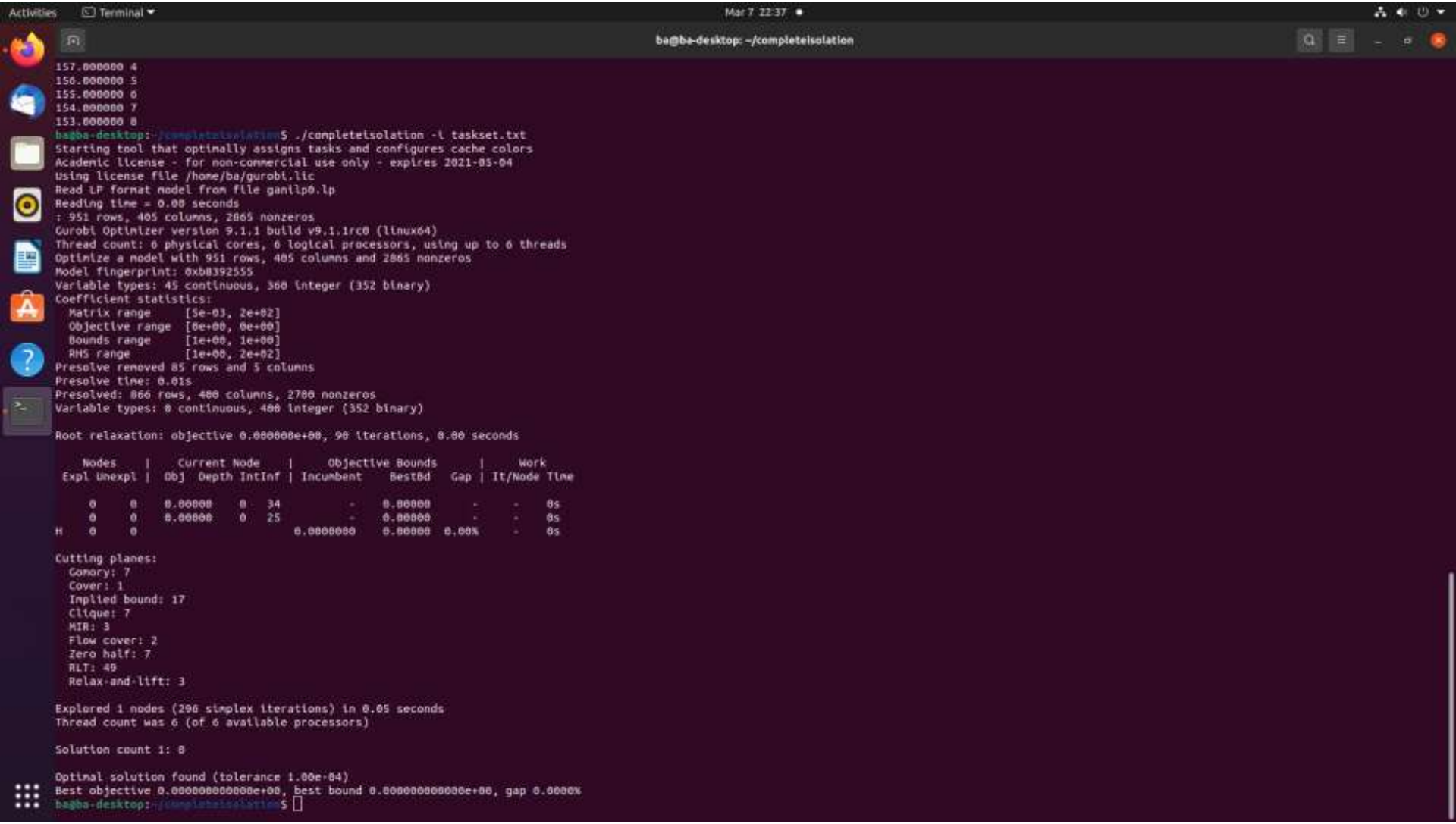


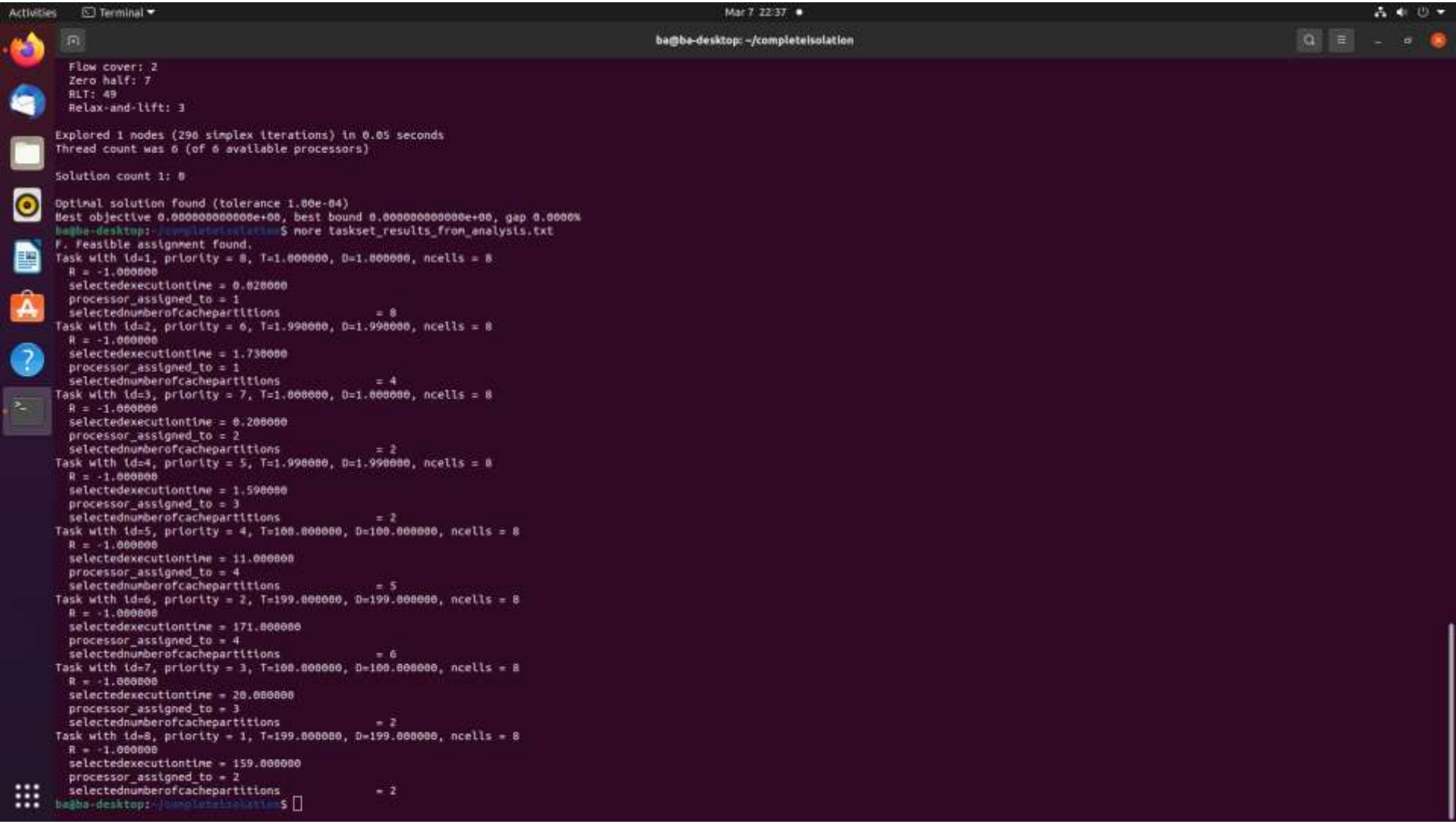












Potential use in a DevOps pipeline

1. In each thread/process, the source code needs to be modified to that before the thread/process starts execution, it reads from a text file to get information. With this information, it calls RK-functions to setup the coordinate cache and bank coloring and also assign it to processor core. Each thread/process has its own file that it reads from in order to get this info to pass to RK-functions. Each thread/process also has an associated file that describes parameters (e.g., period, execution time with different cache colors).

2. When “make” is run, a configuration preparation tool (not yet developed) is invoked. It gathers information about threads/processes from the aforementioned files and puts this info to a single file. This single file is input to the configuration tool (which we already have). Then the configuration tool generates the configuration to a file. A configuration effector tool (not yet developed) reads this file and writes to various files for each thread/process.