



WIKIPEDIA  
The Free Encyclopedia

[Main page](#)  
[Contents](#)  
[Featured content](#)  
[Current events](#)  
[Random article](#)  
[Donate to Wikipedia](#)  
[Wikipedia store](#)

Interaction

[Help](#)  
[About Wikipedia](#)  
[Community portal](#)  
[Recent changes](#)  
[Contact page](#)

Tools

[What links here](#)  
[Related changes](#)  
[Upload file](#)  
[Special pages](#)  
[Permanent link](#)  
[Page information](#)  
[Wikidata item](#)  
[Cite this page](#)

Print/export

[Create a book](#)  
[Download as PDF](#)  
[Printable version](#)

Languages

[Deutsch](#)

 [Edit links](#)

[Create account](#) [Log in](#)

Article [Talk](#)

[Read](#) [Edit](#) [View history](#)

# Fair-share scheduling

From Wikipedia, the free encyclopedia



This article **relies largely or entirely upon a single source**. Relevant discussion may be found on the [talk page](#). Please help [improve this article](#) by introducing [citations](#) to additional sources. *(June 2012)*

**Fair-share scheduling** is a [scheduling algorithm](#) for computer [operating systems](#) in which the [CPU](#) usage is equally distributed among system users or groups, as opposed to equal distribution among processes.

One common method of logically implementing the fair-share scheduling strategy is to recursively apply the [round-robin scheduling](#) strategy at each level of abstraction (processes, users, groups, etc.) The time quantum required by round-robin is arbitrary, as any equal division of time will produce the same results.

This was first developed by Judy Kay and Piers Lauder through their research at Sydney University in the 1980s.<sup>[1]</sup>

## Example [\[edit\]](#)

For example, if four users (A,B,C,D) are concurrently executing one process each, the scheduler will logically divide the available CPU cycles such that each user gets 25% of the whole ( $100\% / 4 = 25\%$ ). If user B starts a second process, each user will still receive 25% of the total cycles, but each of user B's processes will now be attributed 12.5% of the total CPU cycles each, totalling user B's fair share of 25%. On the other hand, if a new user starts a process on the system, the scheduler will reapportion the available CPU cycles such that each user gets 20% of the whole ( $100\% / 5 = 20\%$ ).

Another layer of abstraction allows us to partition users into groups, and apply the fair share algorithm to the groups as well. In this case, the available CPU cycles are divided first among the groups, then among the users within the groups, and then among the processes for that user. For example, if there are three groups (1,2,3) containing three, two, and four users respectively, the available CPU cycles will be distributed as follows:

- $100\% / 3 \text{ groups} = 33.3\% \text{ per group}$
- Group 1: ( $33.3\% / 3 \text{ users}$ ) = 11.1% per user
- Group 2: ( $33.3\% / 2 \text{ users}$ ) = 16.7% per user
- Group 3: ( $33.3\% / 4 \text{ users}$ ) = 8.3% per user

## See also [\[edit\]](#)

- [Completely Fair Scheduler](#)

## References [\[edit\]](#)

- ↑ [http://sydney.edu.au/engineering/it/~judy/Research\\_fair/index.html](http://sydney.edu.au/engineering/it/~judy/Research_fair/index.html)

**Categories:** [Processor scheduling algorithms](#)

This page was last modified on 28 May 2015, at 09:26.

Text is available under the [Creative Commons Attribution-ShareAlike License](#); additional terms may apply. By using this site, you agree to the [Terms of Use](#) and [Privacy Policy](#). Wikipedia® is a registered trademark of the [Wikimedia Foundation, Inc.](#), a non-profit organization.

[Privacy policy](#) [About Wikipedia](#) [Disclaimers](#) [Contact Wikipedia](#) [Developers](#) [Mobile view](#)

