|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Description | Others |
| Row | int | The floor number | Updated in each scrawling process |
| PostTime | String | The time to post | Can be used in updating process to find the range to scrawl |
| PostAmount | int | The total number of posts the user posted | Can be used to get the importance of the post in a way – referring to reply amount |
| Content | String | The content of the post | Using [] to bracket the image source when containing some images |
| personUrl | String | A url pointing to the personal space of the user | Via which we can access the personal space of a certain user |
| personName | String | The nickname of the user |  |
| personID | String | A unique id of the user |  |
| personGender | String | The gender of the user |  |

Apart from the post structure shown in the above table, we will have another five elements to detail the post:

\_\_PostId - the unique id of the post

\_\_Title - the title of the post which at the same time will also specify its parent clubs

\_\_PostTime - which is the post time of the floor one

\_\_PageAmount - the total amount of the pages the post contains

\_\_MaxFloorIndex - the maximum index of the floor the post currently can reach

\_\_url - the URL of the post which can be used to access the post to check what might happen when encountering a problem

To further improve the performance of this scrawling process, I still try to use multi-process instead of multi-thread, which in the end use about one fourth time to do the same job of that in multi-thread or single thread.

The whole structure is about to use different subprocess to handle different clubs when retrieving posts urls; after that different subprocesses will also be adopted to handle different posts, which is rather efficient and as a result hundreds of subprocesses will run at the same time consuming the whole CPU. To make sure this will work right in huge situation, you can set the maximum of subprocesses at the same time, the maximum value is quite relative which means that the exact subprocess number can be around it instead of just its value – the exact reason is unclear to me right now. So far the performance is relatively nice.

Improvements:

1. In PostParser, encountering a big post with lots of pages can be a performance problem in the end. It’s a good idea to use multi-process to handle.
2. In ClubParser, when the club contains lots of posts, there should be a way to further improve the post urls scrawling process; how about using subprocess to handle different page instead of clubs? The multi-process mechanism can be adopted in different levels in different cases, which is the best will depend on the exact situation, I suppose.