多線程中的共享對象

課程

代碼導致了一個奇怪的錯誤。有時會出現錯誤，有時則不會。

這是因為 `translate_markdown_file` 函數，特別是 `translate_front_matter` 函數，可能正在訪問和修改共享的數據結構（如字典或列表），而沒有進行適當的同步。當多個線程同時訪問和修改相同的數據時，可能會導致競爭條件。競爭條件發生時，數據的最終狀態取決於線程執行的不可預測順序。這可能導致數據損壞、意外行為以及您觀察到的間歇性錯誤。

要解決這個問題，您應該避免在線程之間共享可變數據，或者使用適當的同步機制（如鎖）來保護共享數據。在這種情況下，`front_matter_dict` 正在被原地修改，這不是線程安全的。解決方法是創建字典的副本，然後再進行修改。這在代碼中已經完成，但了解為什麼需要這樣做是很重要的。

上下文

```
  with concurrent.futures.ThreadPoolExecutor(max_workers=MAX_THREADS) as executor:
        futures = []
        for filename in changed_files:
            input_file = filename

            for lang in languages:

                print(f"Submitting translation job for {filename} to {lang}...")
                future = executor.submit(translate_markdown_file, input_file, os.path.join(f"_posts/{lang}", c
                futures.append(future)

        for future in concurrent.futures.as_completed(futures):
            try:
                future.result()
            except Exception as e:
                print(f"A thread failed: {e}")
```

之前

```
def translate_front_matter(front_matter, target_language, input_file):
    print(f"  Translating front matter for: {input_file}")
    if not front_matter:
        print(f"  No front matter found for: {input_file}")
        return ""
```

```
try:
    front_matter_dict = {}
    if front_matter:
        front_matter_dict = yaml.safe_load(front_matter)
        print(f"  Front matter after safe_load: {front_matter_dict}")
    if 'title' in front_matter_dict:
        print(f"  Translating title: {front_matter_dict['title']}")
        if not (input_file == 'original/2025-01-11-resume-en.md' and target_language in ['zh', 'fr']):
            if isinstance(front_matter_dict['title'], str):
                translated_title = translate_text(front_matter_dict['title'], target_language)
                if translated_title:
                    translated_title = translated_title.strip()
                    if len(translated_title) > 300:
                        translated_title = translated_title.split('\n')[0]
                    front_matter_dict['title'] = translated_title
                    print(f"  Translated title to: {translated_title}")
                else:
                    print(f"  Title translation failed for: {input_file}")
            else:
                print(f"  Title is not a string, skipping translation for: {input_file}")
        else:
            print(f"  Skipping title translation for {input_file} to {target_language}")
    # Always set lang to target_language

    # Determine if the file is a translation
    original_lang = 'en' # Default to english
    if 'lang' in front_matter_dict:
        original_lang = front_matter_dict['lang']

    if target_language != original_lang:
        front_matter_dict['lang'] = target_language
        front_matter_dict['translated'] = True
        print(f"  Marked as translated to {target_language} for: {input_file}")
    else:
        front_matter_dict['translated'] = False
        print(f"  Not marked as translated for: {input_file}")


    result = "---\n" + yaml.dump(front_matter_dict, allow_unicode=True) + "---"
    print(f"  Front matter translation complete for: {input_file}")
```

```
        return result
    except yaml.YAMLError as e:
        print(f"  Error parsing front matter: {e}")
        return front_matter
```

之後

```
def translate_front_matter(front_matter, target_language, input_file):
    print(f"  Translating front matter for: {input_file}")
    if not front_matter:
        print(f"  No front matter found for: {input_file}")
        return ""
    try:
        front_matter_dict = {}
        if front_matter:
            front_matter_dict = yaml.safe_load(front_matter)
            print(f"  Front matter after safe_load: {front_matter_dict}")

        front_matter_dict_copy = front_matter_dict.copy()

        if 'title' in front_matter_dict_copy:
            print(f"  Translating title: {front_matter_dict_copy['title']}")
            if not (input_file == 'original/2025-01-11-resume-en.md' and target_language in ['zh', 'fr']):
                if isinstance(front_matter_dict_copy['title'], str):
                    translated_title = translate_text(front_matter_dict_copy['title'], target_language)
                    if translated_title:
                        translated_title = translated_title.strip()
                        if len(translated_title) > 300:
                            translated_title = translated_title.split('\n')[0]
                        front_matter_dict_copy['title'] = translated_title
                        print(f"  Translated title to: {translated_title}")
                    else:
                        print(f"  Title translation failed for: {input_file}")
                else:
                    print(f"  Title is not a string, skipping translation for: {input_file}")
            else:
                print(f"  Skipping title translation for {input_file} to {target_language}")
        # Always set lang to target_language

        front_matter_dict_copy['lang'] = target_language
```

```
        front_matter_dict_copy['translated'] = True

        result = "---\n" + yaml.dump(front_matter_dict_copy, allow_unicode=True) + "---"
        print(f"  Front matter translation complete for: {input_file}")
        return result
    except yaml.YAMLError as e:
        print(f"  Error parsing front matter: {e}")
        return front_matter
```