02

2- Linq to entities - Linq paralelo parte 2

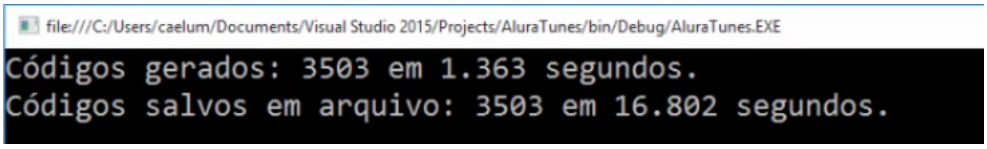
Transcrição

Utilizamos anteriormente o paralelismo para melhorar a performance de uma consulta `LINQ`. Um ponto que ainda deve ser aprimorado é a quantidade de tempo gasto para salvar as imagens!

Antes de otimizar o código, vamos descobrir quanto tempo é gasto atualmente no armazenamento das imagens. Para mensurar esse período, utilizaremos um cronômetro que vamos inserir abaixo do `Console.Writeline()` e nele acrescentaremos uma nova instância, o `stopwatch = Stopwatch.StatNew()`. Ao final, vamos inserir o `stopwatch.Stop()` que indica ao cronômetro que a contagem do tempo deve ser parada. Depois disso vamos copiar a mesma linha do `Console.WriteLine()` que já havíamos utilizado e inseri-la abaixo do `stopwatch.Stop()` e modificaremos de `códigos gerados` para `códigos salvos em arquivo`. Teremos o seguinte:

```
Console.WriteLine("Códigos gerados: {0} em {1} segundos.", contagem, stopwatch.ElapsedMilliseconds /

stopwatch = Stopwatch.StartNew();

foreach (var item in queryCodigos)
{
    item.Imagem.Save(item.Arquivo, ImageFormat.Jpeg);

}
stopwatch.stop();

Console.WriteLine("Códigos salvos em arquivo: {0} em {1} segundos.", contagem, stopwatch.ElapsedMill
```

Ao rodar a aplicação o resultado é o seguinte:



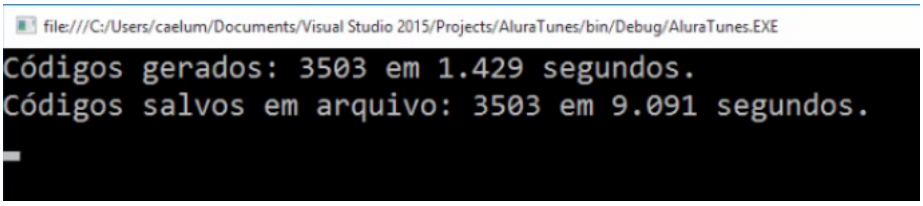
Existe outra técnica para salvar os links! Para aplicá-la, vamos comentar o trecho que vai do `foreach (var item in queryCodigos)` até o `Item.Image.Save()` para que fique desabilitado. Adicionaremos `queryCodigos.ForAll()` e adicionaremos uma expressão `lambda` com o comando de salvar:

```
item => item.Imagem.Save(item.Arquivo, ImageFormat.Jpeg)
```

Observe:

```
queryCodigos.ForAll(item.Arquivo, ImageFormat.Jpeg));

stopwatch.Stop();
```

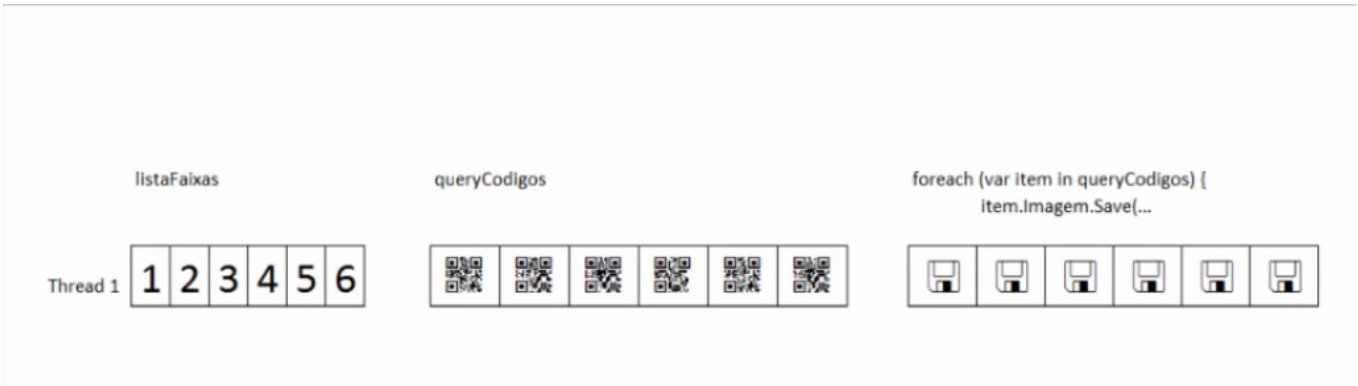
Ao rodarmos a aplicação o resultado é o seguinte:



Isto é, o tempo para salvar as imagens já não é mais `16` segundos, mas sim `9` segundos! Alcançamos uma redução bastante significativa!

Recapitulando: LINQ Paralelo

Observe a seguinte imagem:



Começaremos trabalhando com uma lista de faixas genérica e que continha diversos elementos. Em seguida, avançaremos até uma consulta do tipo `QR Code` que irá converter cada `Id` em um `QR Code` sequencial, um depois do outro. Seguiremos avançando até uma instrução `foreach`. Assim, todas as imagens serão pegas e salvas também de maneira sequencial.

Como aprendemos a realizar operações de maneira sequencial, repetimos o processo. Pegamos os `Id` de diferentes faixas e utilizamos o `AsParallel` para expandir a tarefa de gerar os códigos por diversas *threads*. Pegamos uma única *thread* e expandimos ela em várias e ao final de tudo, tivemos que utilizar o método sequencial para salvar os arquivos.

Um terceiro mecanismo que aprendemos a utilizar é o `ForAll`. Ele pegou os resultados que já estavam paralelizados para reorganizá-los novamente em várias `threads`. Desta maneira, conseguiremos baixar pela metade o tempo de execução do método de salvar uma imagem em arquivo!