Universidade de Coimbra
Faculdade de Ciências e Tecnologia - Departamento de Engenharia Informática
Mestrado em Engenharia Informática - Ramo de Engenharia de Software
Arquitetura de Software - TP1
Autores: Carlos Santos, Gonçalo Pereira, Marco Pereira

Pipe-and-Filter

Trabalho Prático 1

Índice

Em Anexo seguem os diagramas para as decisões implementadas.

Decisões Gerais

Mudanças na *Framework*

Foram realizadas mudanças à "**FilterFramework**" fornecida de modo a permitir várias entradas e várias saídas em cada filtro criado, mantendo, no entanto, a compatibilidade com os filtros existentes. Desta forma, foi possível tornar estes filtros genéricos, criando o número de entradas e saídas pretendidas, sendo que por omissão é criado somente com uma entrada e uma saída. Quando um filtro tem várias entradas e várias saídas tem de se especificar no *connect* que porta do filtro X é conectada ao filtro Y.

É mantido o tipo de dados original na transmissão entre filtros.

Paralelismo

No caso do **Sistema A**, foi testado o recurso a técnicas de **paralelismo** de modo a aumentar a performance nas operações, de filtragem uma vez que, desta maneira, elas são realizadas ao mesmo tempo.

Contudo, na observação dos tempos resultantes desta abordagem quando comparada com a versão **sequencial**, para *inputs* de 100 e 1000 casos, observou-se que a diferença (+/- 2 segundos) não justificaria o investimento no paralelismo com a implementação feita.

Com base nos resultados obtidos para o caso do **Sistema A**, decidiu-se não implementar uma abordagem semelhante nos sistemas seguintes.

Descrição dos componentes:

Splitter: Filtro genérico com 1 entrada e x saídas;

Merger: Filtro genérico com x entradas e 1 saída;

Filtro (temp, height, wildpoints): filtro para modificar os dados com 1 entrada e 1 saída;

Sink: *Printer* com 1 entrada e 1 saída (não usada);

Source: Input com 1 entrada (não usada) e 1 saída;

Sort: Filtro para ordenar a informação com auxilio da *ListNode*. Possui 1 entrada e *1* saída.

Atributos qualidade

Desempenho: foi feito um esforço para o tentar cumprir ao paralelizar ao máximo as operações de filtragem. Contudo, pelo já referido no seu teste no **Sistema A**, o paralelismo não foi escolhido para a versão atual.

Tempos no Sistema A:

- versão sequencial = 55s;
- versão paralela = 58s

Reconfigurabilidade: é cumprido ao criar o maior número de modelos genéricos, para que, caso não se queira realizar uma dada operação ou se queira implementar uma operação diferente, o processo de alteração do fluxo é simples e efetuado rapidamente.

Prós e contras da versão sequencial Vs Paralelismo

Prós Sequencial:

- Menos tempo passado a desenvolver os componentes;
- Sem *Bottleneck* (existente no componente *Merge*), em comparação com o paralelo;
- Componentes mais robustos (sem eventuais problemas de *deadlock*);
- Ganhos de velocidade, pois ao contrário do paralelo não é necessário realizar operações de consistência (no *merge*) de modo a não perder *frames*, após a operação de *split*;
- Mais fácil manter componentes que realizam operações sequencialmente do que paralelismo;
- Possibilidade de inserir um determinado filtro em qualquer parte do fluxo, pois os dados de entrada e saída ao longo do fluxo, estão no mesmo formato;

Contra sequencial:

- O desempenho da arquitetura sequencial será tão bom quanto o desempenho do módulo mais lento;
- Sem mecanismo de *load-balancing*, no caso de uma operação ser mais "*demanding*", não existe a opção de distribuir carga por mais filtros.

Sistema A

1 - Elementos do Sistema A

O sistema A, é composto por

- 4 módulos:
    - 1 *source*
    - 1 altura
    - 1 temperatura
    - 1 *sinc*

2 - Decisões tomadas e porquê

Neste sistema, na sua versão sequencial, os dados da *source* são enviados para os 2 filtros (temperatura, altitude), um a seguir ao outro. Ao passar em cada filtro, são tratados os respectivos dados (conversões) e enviados para o filtro seguinte ate imprimir.

A opção de utilizar paralelismo no tratamento de dados (temperatura e altitude), entretanto colocada de parte pelo já referido no ponto **Paralelismo**, envia os dados recebidos pela *source* para um *splitter* que, por sua vez trata de colocar a informação paralelamente aos filtros existentes. A informação entretanto tratada é colocada num *merge*, que trata de unificar a informação tratada em cada filtro para a mesma *timestamp* da *frame*, enviando de seguida para a impressão no *sink*.

Elementos do Sistema B

O sistema B, é composto por:

- 7 módulos:
    - 1 *source*
    - 1 *splitter*
    - 3 filtros
        - 1 temperatura
        - 1 altura
        - 1 *wildPoint*
    - 2 *sinc*

Decisões tomadas e porque

No que toca à obtenção de dados e tratamento da altura e temperatura, é aproveitado o raciocínio aplicado no **Sistema A**. Após isso, os dados são enviados para um *splitter* que envia todos os *wildpoints* para um *sinc* e o ficheiro para o **filtro dos WildPoints**. Neste filtro os dados são colocados em memória (pois existem dependências entre valores), são modificados os valores da pressão que não estão compreendidos num intervalo dado valor e o fluxo é enviado para o *sinc* final onde a pressão, temperatura e altura estão normalizados.

Elementos do Sistema C

O sistema C, é composto por

- 5 módulos:
    - 2 *sources*
    - 1 *merger*
    - 1 *sort*
    - 1 *sinc*

Decisões tomadas e porque

Neste sistema os dados das *sources* são recebidos aleatoriamente, sendo posteriormente ordenados pela ordem pretendida. A ordenação ocorre num módulo à parte, de modo a manter o requisito de reconfigurabilidade. Caso não se queira ordenar o ficheiro final, basta remover o módulo *sort*, permitindo, também, modificar a ordem do mesmo.

É apenas usada uma thread para receber os valores das *datasources*.

Descrição do estilo da arquitetura

Este estilo de arquitetura é caraterizado pela sua simplicidade e robustez. Consiste num determinado número de componentes (filtros) que transformam ou modificam a informação, antes de passarem a mesma por vários conectores (*pipes*) para outros componentes.

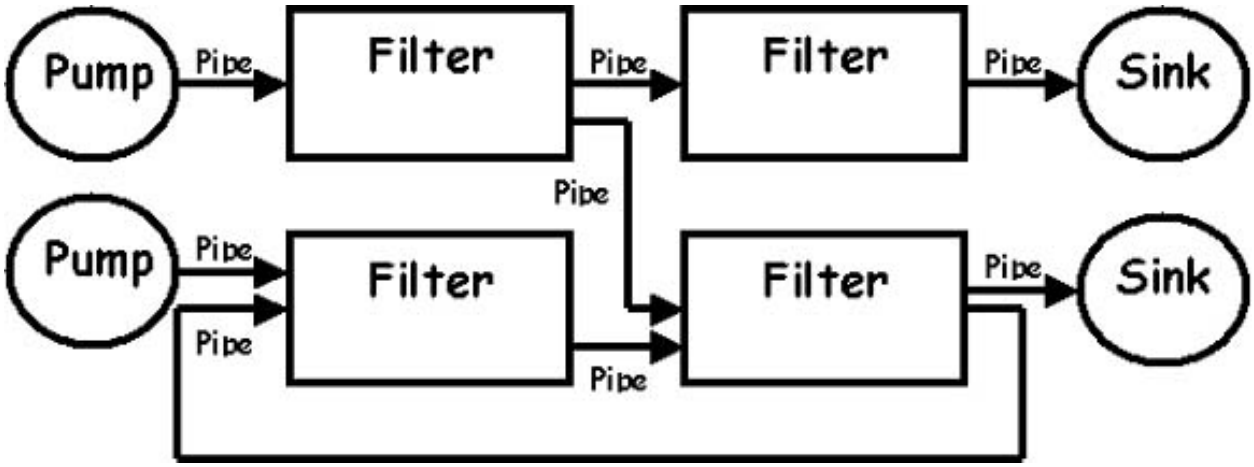

Figura 1 - Arquitetura de um sistema com *Pipe-and-Filter*

Esta arquitetura é constituída por 3 componentes básicos:

1. ***Pump* (produtor):** Produz a informação e coloca-a num porto de *output* que está ligado ao porto de *input* de um *pipe*.

2. ***Filter*:** Recebe a informação a partir de um porto de *input* ligado ao *output* de um *pipe*. Esta é transformada/filtrada e colocada num porto de output ligado ao porto de *input* de um *pipe*. Este componente pode ter várias portas de input (*merge*) ou de output (*splitter*).

3. ***Sink*:** Recebe a informação a partir de um porto de input ligado ao porto de output de um pipe e "consome" a informação.