Instituto de Computação
Universidade Estadual de Campinas

Técnicas Heurísticas de Escalonamento Paralelo em *Workflow*

Leonardo Garcia Tampelini

Este exemplar corresponde à redação final da Dissertação devidamente corrigida e defendida por Leonardo Garcia Tampelini e aprovada pela Banca Examinadora.

Campinas, 28 de março de 2012.

Prof. Dr. Jacques Wainer (Orientador)

Dissertação apresentada ao Instituto de Computação, UNICAMP, como requisito parcial para a obtenção do título de Doutor em Ciência da Computação.

FICHA CATALOGRÁFICA ELABORADA POR
ANA REGINA MACHADO - CRB8/5467
BIBLIOTECA DO INSTITUTO DE MATEMÁTICA, ESTATÍSTICA E
COMPUTAÇÃO CIENTÍFICA - UNICAMP

Tampelini, Leonardo Garcia, 1983-
T153t Técnicas heurísticas de escalonamento paralelo em workflow / Leonardo Garcia Tampelini. – Campinas, SP : [s.n.], 2012.

Orientador: Jacques Wainer.
Dissertação (mestrado) – Universidade Estadual de Campinas, Instituto de Computação.

1. Fluxo de trabalho. 2. Programação heurística. 3. Escalonamento de produção. 4. Simulação (Computadores). I. Wainer, Jacques, 1958-. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

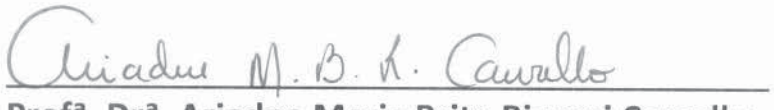
Informações para Biblioteca Digital

Título em inglês: Heuristic scheduling techniques for parallel workflow
Palavras-chave em inglês:
Workflow
Heuristic programming
Tasks scheduling
Computer simulation
Área de concentração: Ciência da Computação
Titulação: Mestre em Ciência da Computação
Banca examinadora:
Jacques Wainer [Orientador]
Debora Pretti Ronconi
Ariadne Maria Brito Rizzoni Carvalho
Data de defesa: 28-03-2012
Programa de Pós-Graduação: Ciência da Computação

TERMO DE APROVAÇÃO

Dissertação Defendida e Aprovada em 28 de Março de 2012, pela Banca examinadora composta pelos Professores Doutores:

Profª. Drª. Debora Pretti Ronconi
DEP / USP



Profª. Drª. Ariadne Maria Brito Rizzoni Carvalho
IC / UNICAMP

Prof. Dr. Jacques Wainer
IC / UNICAMP

Instituto de Computação
Universidade Estadual de Campinas

Técnicas Heurísticas de Escalonamento Paralelo em *Workflow*

Leonardo Garcia Tampelini

Março de 2012

Banca Examinadora:

- Prof. Dr. Jacques Wainer (Orientador)
- Profª. Drª. Debora Pretti Ronconi
  Departamento de Engenharia de Produção - USP/SP
- Profª. Drª. Ariadne Maria Brito Rizzoni Carvalho
  Instituto de Computação - UNICAMP/SP
- Profª. Dr. Eduardo Candido Xavier (Suplente)
  Instituto de Computação - UNICAMP/SP
- Profª. Drª. Renata Mendes de Araujo (Suplente)
  Departamento de Informática Aplicada - UNIRIO/RJ

Resumo

Com a disseminação de tecnologias de gerenciamento empresarial, empresas procuram promover serviços mais ágeis e de maior qualidade. Neste contexto, áreas como gerenciamento de *workflow* vêm contribuindo para uma melhor organização na distribuição de tarefas. A aproximação da área de escalonamento com *workflow* demonstra um grande potencial para atender tais requisitos; porém, uma escassez de trabalhos voltados ao tratamento de estruturas de roteamento paralelas, comumente encontradas em modelos de *workflow*, é perceptível na literatura de escalonamento. Este trabalho tem por objetivo aproximar essas duas áreas apresentando três novas abordagens de escalonamento voltadas à ordenação de casos dentro de estruturas de roteamento paralelas (AND). Para alcançar tal objetivo, um conjunto de simuladores foi implementado representando o ambiente dinâmico de *workflow*, suas incertezas, bem como os diferentes cenários onde estruturas do tipo AND podem ocorrer. O desempenho de tais políticas foi comparado com regras amplamente utilizadas em sistemas de *workflow*, como FIFO (*First In First Out*), EDD (*Earliest Due Date*) e SPT (*Shortest Processing Time*). A análise dos resultados foi efetivada por meio de uma análise de variância (ANOVA) juntamente com o teste de Tukey. Os resultados mostram que é mais vantajoso utilizar técnicas específicas para estrutura de roteamento AND do que apenas aplicar as técnicas mais utilizadas.

Abstract

With the dissemination of business management technologies, companies look for to promoting faster services with higher quality. In this context, areas such as workflow management have contributed to a better organization in the distribution of tasks. The approach between scheduling area and workflow area shows great potential to attend these requirements, but a lack of studies directed to the treatment of parallel routing structures, commonly found in workflow models, is apparent escalation in the literature about scheduling. This work aims to approximate these two areas, presenting three new scheduling approaches, directed to the raging of the cases within routing structures parallel (AND). To reach this objective a set of simulators was implemented, representing the dynamic workflow environment, their uncertainties, as well as the different scenarios where that structures such as AND may occur. The performance of these politics was compared with rules widely used in workflow systems, such as FIFO (First In First Out), EDD (Earliest Due Date) and SPT (Shortest Processing Time). The results show that it is more advantageous to use techniques focused on AND routing structure than only apply the most utilized ones.

Agradecimentos

A Deus pela força, amor, oportunidade e pelo maior presente que ganhei em minha vida, minha família. Pai, Mãe, Irmã e agora a pequena Beatriz, obrigado por todo apoio, carinho e confiança que vocês depositaram em mim.

Ao professor orientador Dr. Jacques Wainer, pelas conversas objetivas e entusiasmo que me ajudaram a ter mais confiança e atingir meus objetivos.

A Isméria, por me tratar como um filho sempre colocando luz no meu caminho.

A minha namorada e companheira Keila Okuda Tavares pela compreensão, alegria, amor, carinho, encorajamento e apoio de todos os momentos.

Aos meus pequenos amigos Mariazinha e Fernandinho, que me demonstram a beleza e a riqueza da energia de ser criança.

Ao meu padrinho, seu Zé, o qual sempre pude contar não importando o tamanho da dificuldade. Obrigado por suas palavras, proteção e apoio.

Ao Leonel, Marlon e Maxiwell, meus amigos e eternos membros da "NossaRepública". Obrigado pelos ótimos momentos que compartilhamos.

A todos os amigos que fiz durante meu mestrado, especialmente ao Erikson e Roberto.

Sumário

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

O ganho de competitividade das organizações está diretamente relacionado com a agilidade em proporcionar serviços ou produtos de qualidade, a custos baixos, de modo que se mantenham padrões de segurança e confiabilidade [15]. Na busca por atender tais requisitos, organizações vêm explorando ferramentas computacionais que possam auxiliar a gerência e o processo de tomada de decisão [33].

Para um melhor aproveitamento do tempo e dos recursos disponíveis, decisões rápidas e dinâmicas são fundamentais, exigindo muita atenção dos administradores no gerenciamento de processos. A fim de otimizar a utilização de recursos disponíveis e diferenciar-se qualitativamente de seus concorrentes, muitas organizações vêm demonstrando grande interesse em Sistemas de Gerenciamento de *Workflow* (WfMs - *Workflow Management System*) [1].

WfMs são utilizados para controlar e melhorar os processos de negócio, assumindo funções mecânicas, burocráticas e otimizando o processo de negócio à medida que ele evolui, permitindo aos usuários se concentrarem em aspectos mais interativos, como a gerencia de recursos humanos [11].

Dentro das atribuições dos WfMs, estão aspectos de roteamento e atribuição dos itens de trabalho aos participantes do *workflow*. Atribuir um item de trabalho a um participante implica em tomar decisões a respeito da ordem de despacho de cada item e, por consequência, de suas prioridades de execução.

Para simplificar tais decisões, a maioria dos WfMs atribuem os itens de trabalho na mesma ordem em que eles chegam ao sistema, ou seja, utilizam uma política FIFO (*First In First Out*), onde o primeiro item a chegar é o primeiro a sair. Alguns sistemas, apresentam todos os itens de trabalho ao seu potencial executor, o qual escolhe ao seu critério um deles, o que equivale à política SIRO (*Service In Random Order*), onde a seleção de um processo se torna um processo aleatório [42]. Apesar da praticidade em aplicar tais métodos, é necessário saber se eles são eficientes.

Para avaliar a eficiência de tais métodos e verificar a possibilidade de aplicar novas técnicas a estruturas específicas, os autores buscaram, na literatura de *workflow*, indicações de como lidar com estes aspectos. Vários trabalhos reconhecem a importância de um melhor gerenciamento no despacho de itens, como [28, 27, 44, 46, 42]. Nesses trabalhos, indicações que a literatura de escalonamento possa trazer benefícios são encontradas.

1.1 Justificativa

Apesar dos benefícios da área de escalonamento serem claros, o mapeamento de problemas de *workflow* para escalonamento não o é, tendo alguns de seus aspectos conflitantes. Como exemplo, pode-se citar a execução paralela de instâncias do mesmo caso em um processo de *workflow*. Esta é uma situação comum e muito importante para sistemas de *workflow*, mas que não encontra representante direto na literatura de escalonamento.

Dada a complexidade de alguns sistemas de *workflow*, que possuem múltiplas estruturas paralelas, e diante a dificuldade de otimizar estruturas paralelas em ambientes dinâmicos, decidiu-se focar o estudo na avaliação e desenvolvimento métodos de escalonamentos voltados a estruturas de roteamento do tipo AND em sistemas de *workflow*.

1.2 Objetivos

O objetivo geral do trabalho é desenvolver políticas de escalonamento especificas para estruturas de roteamento paralelas, comparando seus desempenhos com políticas comumente utilizadas em sistemas de *workflow*. Este objetivo ramifica-se em:

1. Avaliar as estruturas de roteamento paralelas e desenvolver técnicas heurísticas especificas para este fim;

2. Definir um conjunto de cenários que represente os aspectos mais comuns encontrados em sistemas de workflow;

3. Desenvolver um simulador de *workflow* capaz de representar um ambiente dinâmico, onde casos chegam enquanto outros já estão sendo processados e alguns já estão sendo concluídos, e representar incertezas quanto ao tempo de execução dos casos.

O desempenho aqui mencionado diz respeito às métricas utilizadas neste trabalho, moldadas para capturar aspectos importantes de processos de workflow, como porcentagem de casos atrasados, e o tempo de processamento destes casos, no sistema.

1.3 Organização do Trabalho

Este trabalho é dividido em 7 capítulos, sendo o primeiro esta introdução, que apresenta os conceitos gerais, a justificativa e os objetivos deste trabalho.

No capítulo seguinte (Capítulo 2) é feita uma revisão da principal literatura sobre *workflow* e as metodologias utilizadas na sua abordagem. A atenção é focada sobre problemas que são semelhantes ou têm interesse para o problema em análise

O Capítulo 3 define os principais conceitos das áreas de *workflow* e escalonamento. Também no Capítulo 3, são apresentadas, as principais dificuldades envolvidas na junção dessas duas áreas.

O Capítulo 4 constrói o núcleo fundamental da dissertação, onde é descrito o problema de sincronismo em estruturas paralelas. É lá que são apresentadas as heurísticas propostas para a redução do tempo de processamento e número de casos atrasados em estruturas de roteamento AND.

No Capítulo 5, uma descrição do ambiente construído para executar as simulações, os parâmetros utilizados em cada um dos testes e o método de avaliação dos resultados.

No Capítulo 6, os resultados obtidos neste trabalho são apresentados, e uma discussão sobre suas implicações no sistema é realizada.

Por fim, o Capítulo 7 traz as conclusões e sugestões de trabalhos futuros a este estudo.

Capítulo 2

Revisão Bibliográfica

Este capítulo mostra a revisão blibliográfica feita para esta pesquisa, que culminou no estudo e desenvolvimento das estratégias de escalonamento voltadas a estrutura de roteamento paralela em *workflow*.

2.1 Escalonamento em *Workflow*

Trabalhos anteriores reconhecem a existência de aspectos temporais e de escalonamento em sistemas de *workflow*. Mans *et al.* [27], preocupam-se com a maximização do uso de recursos e com a redução dos tempos de construção dos processos em um ambiente hospitalar, utilizando para tal fim um conceito de escalonamento por calendários de alocação, sendo atribuído um calendário a cada recurso (médicos e enfermeiras) e requisitante (paciente).

Zhao e Stohr [46] desenvolvem um algoritmo para prever os tempos de execução de casos em um sistema de *workflow*, sendo utilizado como base de um processo de alocação de recursos. O algoritmo se baseia diretamente em técnicas de escalonamento, utilizando um sistema de funções de recompensas para guiar a execução dos casos, recompensando individualmente os executores caso eles executem uma tarefa específica, ao invés de outras.

No artigo apresentado por Marazakis e Nicolaous [28], os autores seguem uma linha similar à que é seguida neste trabalho. Marazakis e Nicolaous utilizam métricas para avaliar o desempenho de políticas de escalonamento em um sistema de gerenciamento de *workflow* dinâmico, considerando a existência de filas e utilizando a simulação como ambiente de análise.

Esses trabalhos apresentam muitos aspectos de otimização similares com esta dissertação, porém os autores não identificam e nem abordam o problema de escalonamento paralelo como sendo um caso específico.

2.2 Restrições de Tempo

Muitos autores conduziram pesquisas nas áreas de gerenciamento de tempo, *time reasoning*, e satisfação de restrições de tempo em sistemas de *workflow*. Nestes trabalhos, o principal foco está em definir restrições (*constraints*) para os tempos de processamento dos casos nos processos, e em procurar algoritmos e soluções para escalonar tais casos sem violar tais restrições. Nesta linha de pesquisa, Eder *et al.* [17] e Bettini *et al.* [5] investigam a especificação e satisfação de restrições de tempo, em sistemas de *workflow*. Ambos estendem sua definição de workflow para suportar restrições de tempo e provêem algoritmos para verificar se tais restrições podem ser satisfeitas e alertar para possíveis violações dessas restrições.

Combi e Pozz [13] realizam uma classificação dos recursos existentes definindo restrições de custo (por exemplo, o custo de execução por hora de um recurso) mantendo as restrições temporais (como: *dead line* e disponibilidade). Os autores buscam atribuir os recursos as tarefas por meio de um sistema de adequação evitando desperdiçar um recurso muito custoso com tarefas pouco expressivas.

Nesta linha de pesquisa, a métrica de avaliação é apenas a satisfação ou não das restrições. Qualquer resultado que satisfaça as restrições é considerado válido. No caso deste trabalho, não se buscam soluções que satisfaçam restrições, mas soluções que otimizem métricas numéricas definidas para os cenários. Tais métricas são definidas sem relação com restrições.

2.3 Simulação e Wokflow

Apesar da maioria dos trabalhos utilizarem a simulação apenas como meio de validação de *workflow*, alguns trabalhos se concentram no desenvolvimento de técnicas específicas para a captura e representação de situações reais. Normalmente, tais publicações se concentram na modelagem de aspectos estatísticos ([23], [24]) ou no desenvolvimento de linguagens específicas de simulação ([22]).

O trabalho desenvolvido por Rozinat *et al.* [34] apresenta um sistema de simulação de *workflow* voltado ao apoio de decisão. Para avaliar o estado real do *workflow* e permitir fazer inferências sobre o comportamento futuro do sistema, um histórico dos estados é armazenado. Esses dados são utilizados para auxiliar na tomada de decisão sobre alocação de recursos. Apesar da divergência de objetivos, esse trabalho foi útil na definição do modelo de comportamento dos recursos.

2.4 Workflow em *Grids*

O paradigma de grade computacional tornou-se amplamente utilizado no processamento de grande quantidade de dados sobre um sistema geograficamente distribuído. Um *Grid*, ou uma grade computacional, é um ambiente dinâmico, heterogêneo e compartilhado. É uma área bastante pesquisada recentemente, envolvendo inclusive aplicações de *workflow* científico de grande escala e gerenciamento de multiplos *workflows* [45], [6].

Zhao e Sakellariou [45] analisam alguns métodos de composição de *workflows* para realizar o escalonamento de múltiplos *workflows*, além de apresentar duas políticas para obter justiça entre os workflows escalonados (decidir qual dos *workflows* deve ser escalonado primeiro). Os autores analisam o comportamento de duas heurísticas de escalonamento para um único *workflow*, utilizando os métodos e políticas propostas para que os múltiplos *workflows* sejam escalonados por essas heurísticas. Os autores concluem que é possível manter justiça entre os *workflows* enfatizando a importância de soluções heurísticas neste tipo de escalonamento.

O trabalho de Ayyub e Abramson [36] propõe um escalonador dinâmico para *workflows* em grades modelando a comunicação entre os *workflows* que estão executando. O desempenho das tarefas e das aplicações como um todo é definido em relação a quão bem elas podem se comunicar e quanto essa comunicação afeta seu desempenho final. Neste caso, o trabalho difere desta dissertação já na concepção, visto que aqui o objetivo principal não é melhorar a comunicação entre as tarefas de um processo, mas melhorar o desempenho do processo pela reordenação da execução dessas atividades.

2.5 Definição de Recursos

Alguns autores focam-se em otimizar a distribuição de recursos, procurando manter apenas os recursos necessários para que os objetivos de negócio sejam alcançados. Nesta linha de pesquisa, Laguna e Marklund [25] apresentam uma extenso trabalho sobre a análise de processos de negócio por meio de critérios quantitativos e qualitativos, teoria de filas e foco na simulação como uma ferramenta para entender e melhorar tais processos. Reijers [32] apresenta técnicas para avaliar o desempenho de processos de *workflow* e discute a alocação ótima de recursos em *workflows*. Hee *et al.* [20] propõem um um método de alocação de recursos em um processo de negócio, o qual chamam de *marginal allocation*, para tentar otimizar o *sojourn time* (o tempo de processamento) dos casos no sistema. Eles comparam o método proposto com o princípio de Goldratt [18]. Neste trabalho, por sua vez, o conjunto dos recursos em um processo é fixo, e a discussão gira em torno de como reordenar os casos presentes nas filas de tais recursos para atingir melhoras de desempenho.

2.6 Otimização de Estruturas de Roteamento

Tramontina [40] apresenta um estudo aprofundado do comportamento de técnicas de escalonamento em sistemas de *workflow* e propõe uma metodologia de aplicação destas técnicas de escalonamento em cenários de *workflow* complexos. A construção da metodologia é baseada na análise do comportamento de técnicas de escalonamento em cenários básicos de *workflow* como: sequência, ou, iterativo e paralelo. Apesar da simplicidade dos cenários iniciais eles levam em conta três características importantes: eles são dinâmicos, possuem incertezas quanto ao tempo de processamento das tarefas em suas atividades, e também possuem incertezas quanto à rota que tais tarefas seguem nos desvios condicionais dos processos.

Para cada cenário básico foram usadas tanto técnicas locais de escolha de tarefas quanto algoritmos genéticos, que possuem uma visão global do problema. O conjunto de informações colhidas a partir dos cenários básicos foi utilizado para decidir qual política é mais eficiente em cada um dos cenários básicos. De posse dessas informações, um cenário mais complexo é dividido em partes básicas e diversas políticas podem ser aplicadas em um único cenário complexo, aumentando a eficiência do *workflow*.

Uma das dificuldades encontradas no trabalho do Tramontina [40] foi o escalonamento de estruturas de roteamento do tipo AND. A inserção de estruturas do tipo AND em cenários mais complexos contribuiu significantemente para uma redução do desempenho dos algoritmos de escalonamento propostos pelo autor, enfatizando a necessidade de um estudo mais detalhado de tais estruturas.

Apesar das diferenças encontradas nos trabalhos citados, eles podem ser vistos como complementares ao trabalho aqui proposto. Tanto os modelos propostos por aqueles autores podem ser refinados utilizando-se os resultados deste trabalho, quanto este pode vir a incorporar aspectos daqueles.

Capítulo 3

Workflow e Escalonamento: Conceitos

O gerenciamento de divisão de tarefas aproxima a área de *workflow* com a área de escalonamento, propiciando o desenvolvimento de novas técnicas que buscam auxiliar o melhor aproveitamento dos recursos disponíveis.

Este capítulo aborda conceitos básicos de *workflow* e de escalonamento, estabelecendo um conjunto de definições que servem de referência para o restante deste trabalho.

3.1 *Workflow*

A ampla aplicabilidade de *workflow* em diversas áreas, gera na literatura uma grande variedade de definições ([3, 9, 35, 37, 38]). Porém, todas são apoiadas em uma mesma linha: *workflow* lida com a execução de processos de negócios e é responsável pela gerência de fluxo de informações.

Dentre as definições encontradas na literatura, a utilizada pela *Workflow Management Coalition* (WfMC) [12] se destaca por sua generalidade. Segundo WfMC, *workflow* é a automação de procedimentos, em parte ou integral, através da qual documentos, informações ou atividades são passadas de um participante ao outro, respeitando um conjunto definido de regras, tendo como finalidade atingir ou contribuir para com os objetivos de uma organização.

A descrição do conjunto de atividades que fazem parte do processo de negócio e a relação de ordem existente entre as atividades é inerente aos modelos de *worklfow*. Segundo Aalst *et al.* [42], a semelhança entre *workflow* e processos de negócio, muitas vezes, faz com que alguns autores utilizem esses dois termos como sinônimos. Apesar de a literatura mostrar claramente os benefícios de sistemas de *workflow* em otimizar e organizar o processo de negócio, de acordo com [42], por falta de um padrão de modelagem, muitas

organizações ficam relutantes em utilizar ferramentas de gerenciamento de *workflow*.

Com a finalidade de desenvolver e aumentar a utilização das tecnologias de *workflow*, a *Workflow Management Coalition* procura estabelecer padrões e terminologias comuns, também contribuindo para o compartilhamento de conhecimento na área.

A WfMC desenvolveu um conjunto de documentações que definem um modelo para sistemas de *workflow* [11], [12]. O modelo, denominado de *Workflow Reference Model*, identifica as características, terminologias e componentes dos WfMCs, e ainda habilita especificações individuais para serem desenvolvidas dentro deste contexto.

3.1.1 Definições Iniciais

A seguir serão apresentadas, de acordo com a documentação divulgada pela WfMC, algumas definições importantes para o entendimento do contexto no qual se encontra inserido um sistema de *workflow* [11], [12]:

- **processo de negócio:** conjunto de um ou mais procedimentos ou atividades ligadas que, coletivamente, têm um objetivo comum; no contexto de uma estrutura organizacional, o processo de negócio é definido por meio de funções e regras de relacionamento; as atividades podem ser executadas de forma sequencial ou paralela, sendo permitida a inclusão de regras de precedência e interdependência entre as atividades.

- **definição de processo:** representação de um processo de negócio que permita a manipulação automatizada; a definição de um processo consiste em uma rede de atividades e seus relacionamentos, contendo critérios para seu início e término, além de informações sobre os recursos associados, ferramentas necessárias para sua realização e subprocessos necessários para sua finalização.

- **atividade e recurso:** unidade lógica e indivisível de trabalho pertencente a um processo, a qual deve ser executada por um recurso; este recurso, por sua vez, refere-se a uma pessoa ou algo que possa se responsabilizar pela conclusão de uma tarefa; uma atividade pode ser executada de forma automatizada ou manual.

- **instâncias ou caso:** ocorrências de uma determinada atividade dentro da execução de um processo; ao iniciar um processo, pode-se inferir que uma nova instância de execução foi criada, a qual possui atributos referentes ao fluxo de dados do processo; o fato de uma instância ter utilizado um conjunto de caminhos até a sua conclusão não implica que novas instâncias percorrerão os mesmos caminhos, pois cada instância é independente das demais e sua execução depende apenas de seus atributos.

- **item de trabalho:** interação entre uma instância do processo (caso) e uma atividade pela qual esta instância deverá passar; no momento em que um determinado caso requer a execução de uma atividade, um item de trabalho vinculado à atividade é criado, o qual será executado por um recurso específico; a execução de uma atividade é na verdade a execução de um item de trabalho.

Vale ressaltar que os termos referentes à atividade e a execução divergem um pouco da nomenclatura inglesa. O termo *task* é utilizado para designar uma atividade, enquanto *activity* é utilizado para designar a execução da atividade.

3.1.2 Roteamento e Alocação de Itens de Trabalho

O fluxo de controle permite a sincronização do processo de execução das atividades. Conexões são utilizadas para representar as interações das atividades. Esta representação permite uma melhor visualização dos caminhos que cada caso pode seguir em um sistema de *workflow*. Porém, o roteamento dos casos só será conhecido durante a execução. Existem quatro situações possíveis de roteamento com as quais o sistema de *workflow* deve lidar [42]:

- *roteamento sequencial*: é a situação mais simples, ocorre quando há uma única linha de precedência, ou seja, os casos devem ser executados em sequência; a Figura 3.1 representa este fluxo.

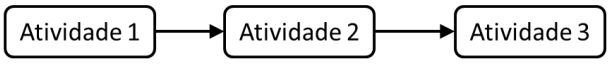


Figura 3.1: Exemplo de fluxo sequencial de atividades.

- *roteamento paralelo* (AND): ocorre quando uma linha de fluxo se divide em duas ou mais linhas, permitindo que atividades paralelas executem de forma simultânea e independente; este ponto de divisão de fluxo é denominado de AND-*Split*, e o ponto onde os fluxos divididos de um AND convergem em uma única linha de fluxo define um AND-*Join*; tal fluxo é representado na Figura 3.2.

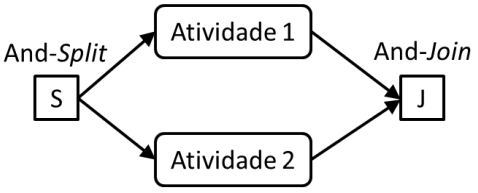


Figura 3.2: Atividades obedecendo a um fluxo paralelo.

- *roteamento condicional* (OR): acontece quando a linha de fluxo divide-se em mais de um caminho a ser tomado; neste ponto, denominado de OR-*Split*, uma decisão sobre qual atividade deve ser executada deve ser tomada; esta escolha depende apenas dos atributos de cada caso; o ponto no qual uma atividade que tem um número de alternativas converge para uma única escolha é chamado de OR-Join; o fluxo condicional é mostrado na Figura 3.3.

Figura 3.3: Elementos OR-Split e OR-Join na representação de fluxo condicional.

- *roteamento iterativo* (LOOP): representa atividades que devem ser executadas repetidamente até encontrar a condição satisfatória de saída; atividades que dependem de uma validação final para serem concluídas são bons exemplos da utilização de LOOP; a Figura 3.4 exemplifica o fluxo iterativo.

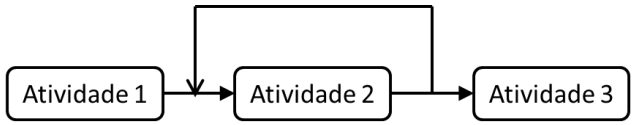


Figura 3.4: Representação de um fluxo iterativo de atividades.

O processo realizado pelo sistema de *workflow* para distribuir os itens de trabalho aos recursos define o processo de alocação (escalonador). Esta designação de trabalho deve obedecer a certas restrições: como não alocar serviço a alguém não autorizado, definir se o recurso terá autonomia para decidir a ordem de execução dos itens de trabalho disponíveis a ele, e manter as informações sobre a carga de trabalho dos recursos.

3.1.3 Sistema de Gerenciamento de Workflow

Sistema que define, cria e gerencia a execução de *workflows* por meio do uso de sistemas computacionais, sendo executado em uma ou mais máquinas de *workflow*, capacitado para interpretar a definição de um processo de negócio, interagir com os participantes e, quando necessário, fazer uso de ferramentas auxiliares [11] [12].

É importante ressaltar que a tecnologia envolvida nos sistemas de gererenciamento de *workflow* não modifica o processo de negócio ou os fluxos internos de um modelo. Os

sistemas de gerenciamento de *workflow* não consertam processos, apenas controlam seu fluxo de execução e, por meio dos dados internos utilizados para o controle, permitem que sejam detectados pontos críticos que devem ser verificados por seus responsáveis para a ação corretiva.

A decisão de mudança do modelo é externa ao sistema e não se deve confundir sistemas de gerenciamento de *workflow* com sistemas que avaliam situações e tomam decisões de projeto.

O conjunto de situações possíveis que o sistema de gerenciamento de *workflow* pode assumir é previamente definido durante a fase de análise e mapeamento do processo.

Os benefícios trazidos pelo uso de sistemas de gerenciamento de *workflow* tornam-se mais visíveis quando o número de atividades a serem coordenadas aumenta. Um mesmo processo de negócio pode ter um ciclo de vida que dure alguns minutos, dias, ou até mesmo meses e anos, dependendo da complexidade e da duração das atividades que o compõem.

A utilização de sistemas de gerenciamento, ou simplesmente WfMS, facilita a coordenação de atividades, permitindo uma melhor visão do estado do processo.

Para que uma ferramenta seja considerada um WfMS, algumas características básicas devem estar presentes. A padronização dos requisitos de um sistema de gerenciamento de *workflow* é definida pela WfMC em [11]. Uma descrição sucinta dos principais funcionalidades é apresentada a seguir:

- **definição do processo:** capacidade de lidar com a definição e modelagem dos processos de *workflow* e suas atividades; estabelece a definição completa de processo;
- **execução de processo:** permitir a iteração com usuários e outras aplicações utilizadas para o processamento das atividades contidas na definição do processo;
- **gerenciamento do processo:** conjunto de funções e aplicações utilizadas para o gerenciamento da execução das atividades dentro do ambiente operacional, garantindo a ordem correta de execução de acordo com o conjunto de regras definidas durante a fase de definição do processo; conjunto de serviços diretamente ligado a interação com o usuário.

Segundo a WfMC, a relação entre essas funcionalidades, representada pela Figura 3.5, define a estrutura que um projeto de automação de *workflow* deve seguir.

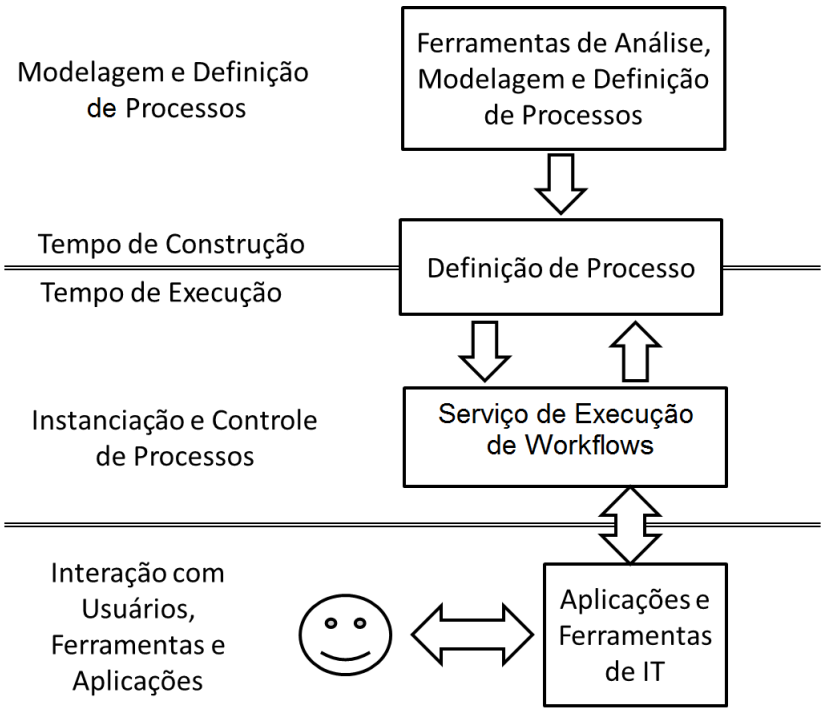


Figura 3.5: Estrutura básica de um sistema de gerenciamento de workflow proposta pela WfMC.

3.2 Escalonamento

O estudo do escalonamento começou a tornar-se importante no início do século XX, sendo Henry Laurence Gantt um dos principais pioneiros na área. O problema abordado inicialmente foi o escalonamento na produção industrial, evoluindo durante os anos através da formalização dada por formulações em programação dinâmica e programação inteira, e pela hierarquização da complexidade dos problemas de escalonamento [76]. Mais tarde, com a popularização dos computadores, o desenvolvimento de sistemas computacionais de escalonamento passou a ser estudado, sendo um foco de pesquisa atualmente. Portanto, o escalonamento é um problema genérico que é encontrado nas mais variadas áreas, sendo definido de maneiras diferentes de acordo com o problema abordado.

Segundo Pinedo [31], escalonamento é um processo de tomada de decisão que se preocupa com a alocação de recursos limitados para tarefas ao longo do tempo, e que possui como meta a otimização de uma ou mais funções objetivo. O gerenciamento de recursos limitados envolve lidar com prioridades, estabelecidas por meio de um conjunto de regras previamente definidas.

De um modo geral, nos problemas de escalonamento assume-se a necessidade de exe-

cutar certo número de processos, ou tarefas, cada um dos quais descritos em uma dada sequência de operações, utilizando para tal fim os recursos disponíveis. O processamento de uma operação requer o uso de uma máquina particular durante certo intervalo de tempo. Neste contexto, a função objetivo serve para avaliar o custo de cada solução possível, permitindo uma busca pelo menor custo [14].

Porém, satisfazer as funções objetivo nem sempre é uma tarefa viável. Escalonamento de tarefas é um problema NP-Completo [31], o problema de otimização associado é NP-Difícil e não possui algoritmo conhecido que gere soluções ótimas em tempo polinomial de forma determinística. Com isso, algumas técnicas são utilizadas na tentativa de aproximar a solução ótima com algoritmos de complexidade de tempo polinomial. Algumas técnicas e suas características são listadas abaixo:

- **Regras simples:** nesta classe, regras simples de prioridade são utilizadas. As informações utilizadas para definir as prioridades baseiam-se em atributos facilmente elicitados; entre as os atributos mais comuns utilizados estão: os tempos de processamento, datas de vencimento, e tempos de chegada; dentre as políticas mais utilizadas temos:

  FIFO - *First In First Out* : repassa ao recurso o caso que está há mais tempo no sistema; a vantagem está em sua simplicidade, não exigindo um pré-processamento para decidir qual é o próximo caso que deve ser processado; apesar da sua simplicidade, está política é a mais utilizada nos sistemas de *workflow* atuais para despachar trabalho aos seus participantes [42].

  SIRO - *Service In Random Order* : seleciona os casos em ordem aleatória; situação comum em sistemas de *workflow*, onde o próprio recurso escolhe, dentre os casos disponíveis, qual irá executar.

  EDD - *Earliest Due Date* : seleciona os casos pela ordem de seus prazos; quem possui o menor prazo é executado primeiro; regra eficaz quando se deseja minimizar o número de casos atrasados em sistemas pequenos.

  SPT - *Shortest Processing Time* : seleciona os casos na ordem de seu tempo de processamento no recurso em questão; o caso que possui o menor tempo de processamento é executado primeiro; para sistemas $1||\sum C_j$, ou seja, com uma atividade, onde se deseja minimizar o somatório dos tempos de finalização dos casos, a política SPT é ótima.

- **Heurísticas:** podem apresentar baixa complexidade e execução rápida, porém geralmente não fornecem garantias em relação à qualidade da solução obtida;

- **Meta-heurísticas:** tempo de execução depende da condição de parada (por exemplo, número de iterações) imposta pelo programador. Para alcançar soluções de boa qualidade, geralmente precisam de um tempo de execução maior que as heurísticas.Também não fornecem garantias sobre a qualidade da solução: ótimos locais são frequentemente a solução final;

- **Programação linear:** tempo de execução e qualidade da solução dependem do relaxamento de restrições e da redução do número de variáveis;

- **Algoritmos de aproximação:** a obtenção de algoritmos de aproximação com baixa complexidade e com fator de aproximação razoável é difícil para problemas genéricos, envolvendo o uso de ferramentas do desenvolvimento de algoritmos exatos e a obtenção de limitantes justos [43]. Algoritmos de aproximação fornecem garantia quanto à qualidade da solução, apresentando limitantes em relação à solução ótima. Quanto mais geral o problema, mais difícil é a obtenção de uma aproximação satisfatória.

Apesar de complexo e custoso o escalonamento é fundamental para atingir bom desempenho na execução de processos em um sistema de *workflow* do tipo heterogêneo. Nesse sistema, diversas variáveis são consideradas para simular ambientes reais de escalonamento (a citar: variáveis associadas tempos de computação, a comunicação e custo de execução). Apesar da gama de variáveis que podem ser utilizada para definir o escopo de um processo de escalonamento, tais problemas podem ser classificados a partir de quatro tipos de informações básicas [14]:

1. os tipos de trabalhos a serem processados;

2. as características dos recursos e a quantidade de recursos disponíveis para execução de trabalho;

3. os requisitos de execução dos trabalhos e, as regras que definem uma alocação a um determinado recurso (como, prioridade e prazos), e;

4. os critérios pelos quais o escalonamento será avaliado.

Além da descrição das características de um problema de escalonamento, uma metodologia para catalogar problemas de escalonamento se torna necessária. Graham *et al.* [19] em 1979 propôs uma classificação para problemas de escalonamento, posteriormente complementada e estendida por outros autores ([7], [16]), composta por uma notação de três campos, $\alpha \,|\, \beta |\, \gamma$. O termo α descreve o ambiente das máquinas que processam os trabalhos, e possui um único valor. O termo β descreve características de processamento

dos trabalhos e pode conter nenhum, apenas um, ou múltiplos valores. Por último, γ denota a função objetivo a ser minimizada, podendo receber um ou mais valores.

Para uma descrição mais detalhada é necessário estabelecer algumas nomenclaturas. O número de trabalhos (*jobs*) a serem executados é aqui considerado finito, e representado por n. A quantidade de máquinas disponíveis para executar o conjunto de *jobs* J é definida com m. O processamento de um *job* j em uma máquina i é representado por um par (i, j). Para um determinado *job* j, os parâmetros que o definem são:

- **processing time (p_i, j):** representa o tempo gasto para que o *job* j seja processado na máquina i; omitindo-se o segundo índice i, retira-se a dependência entre o tempo de execução e máquina executora; utilizado quando todas as máquinas são idênticas;
- **release date (r_j):** tempo que representa o momento em que um *job* j chega ao sistema e torna-se disponível para o processamento; um *job* j só pode ser processado a partir deste tempo;
- **due date (d_j):** momento em que o *job* j deve estar pronto; utilizado para definir prazos de entrega;
- **weight (w_j):** atribuição de importância ao *job* j.

3.2.1 Ambiente de Processamento (α)

O ambiente de processamento é caracterizado por uma cadeia de caracteres constituída de dois parâmetros $\alpha = \alpha_1\alpha_2$, sendo atribuído o símbolo $\emptyset$[1] para representar vazio. O parâmetro $\alpha \in (\emptyset, P, Q, R, O, F, FF, J)$ caracteriza o tipo de máquina e possui o seguinte significado:

$\alpha = \emptyset$**:** caso onde apenas se tem uma única máquina ou processador;

$\alpha = P$**:** denota a existência de máquinas idênticas e paralelas; o job j requer apenas uma operação e pode ser processado em qualquer uma das máquinas;

$\alpha = Q$**:** representa a existência de máquinas em paralelo, porém distintas no quesito velocidade de execução v; este ambiente também é referenciado com o nome de *máquinas uniformes*;

$\alpha = R$**:** conhecida como *unrelated machines in parallel*, representa uma generalização do ambiente anterior; existem máquinas em paralelo, porém a velocidade da máquina depende do *job* a ser processado nela, ou seja, a máquina i pode processar o *job*

[1] O símbolo $\emptyset$ significa "vazio" e pode ser omitido, segundo a classificação de Pinedo [31].

j com velocidade v_i, j, onde o tempo de processamento p_i, j do *job* j é dado por $p_j/v_i, j$;

$\alpha = O$**:** ou modo *open shop*, requisita que cada *job* j deva ser processado novamente em cada uma das m máquinas, não tendo restrições de percurso, ou seja, o escalonador tem autonomia de decisão da rota; vale ressaltar que o tempo de execução, para alguns casos, pode ser igual à zero;

$\alpha = F$**:** neste modo, também nomeado de processamento dedicado *flow shop*, cada *job* deve ser processado em todas as máquinas m e seguir uma mesma ordem de processamento;

$\alpha = FF$**:** denominado de *flexible flow shop*, representa uma generalização do *flow shop* e do ambiente com máquinas paralelas; ao invés de m máquinas em série, existem c centros de trabalho agrupando máquinas idênticas que funcionam em paralelo; cada trabalho tem que ser processado no centro de trabalho 1, 2 e assim por diante. Um trabalho é processado em uma máquina de um centro de trabalho, mas todas as máquinas deste centro poderiam processá-lo da mesma forma.

$\alpha = J$**:** em um *job shop*, cada *job* tem uma rota pré-determinada para seguir; cada trabalho somente é escalonado uma única vez; cada máquina pode processar um único trabalho por vez, sendo que nenhuma máquina pode ser liberada antes de finalizar a operação em processo.

O segundo parâmetro, α_2, representa o número de máquinas do ambiente de processamento. Desse modo, somente valores inteiros e positivos podem ser atribuídos a α_2. Caso nenhum valor seja atribuído a α_2, ou seja, $\alpha_2 = \emptyset$, então o número de máquinas no sistema será variável. No caso onde existe somente uma única máquina processante, temos $\alpha_1 = \emptyset$ e $\alpha_2 = 1$.

3.2.2 Características dos *Jobs* e Recursos (β)

O campo β da definição de problemas de escalonamento indica as características de execução incluídas nos recursos para a execução dos *jobs*. O campo β representa um conjunto composto de oito elementos, $\beta = \beta_1, \beta_2, \beta_3, \beta_4, \beta_5, \beta_6, \beta_7, \beta_8$. Os possíveis valores desse termo, bem como uma descrição de seus significados, são apresentados na sequência:

$\beta_1 \in (\emptyset, pmtn)$**:** este primeiro elemento refere-se à permissão de preempção de *jobs* pelos recursos; um recurso preemptivo pode interromper momentaneamente a execução de um determinado *job* j antes de seu término, liberando o recurso e retornando ou

transferindo sua execução *job j* em um momento apropriado; o valor $\emptyset$ indica que a preempção não é permitida e *pmtn* que a preempção é permitida;

$\beta_2 \in (\emptyset, res)$**:** representa a existência de requisitos na utilização de recursos; para o valor res, cada *job j* requer o uso de r_hj unidades de $R_h(h = 1, ..., s)$ para sua execução;

$\beta_3 \in (\emptyset, prec, uan, tree, chains)$**:** indica a existência de uma relação de precedência entre os *jobs*; o valor $\emptyset$ representa a não existência de precedência; *prec* representa restrições de precedência genéricas, estipuladas por um *DAG (Directed Acyclic Graph)*; *uan* indica restrições de precedência do tipo *uniconnected activity networks*; *tree* indica restrições de precedência em forma de árvore; *chains* indica restrições de precedência que formam um conjunto de cadeias;

$\beta_4 \in (\emptyset, r_j)$**:** estipula uma data mínima de início para a execução dos *jobs*; o parâmetro β_4 pode representar duas situações: a primeira, $\beta_4 = \emptyset$, indica que todas as tarefas podem iniciar no tempo zero, e a segunda, $\beta_4 = r_j$, onde as datas de início de cada *job j* variam de acordo com a tarefa; caso os *jobs* possuam prazos, nada é especificado; a função objetivo, do campo γ, será suficiente verificar a relevância deste aspecto no problema

$\beta_5 \in (\emptyset, brkdwn)$**:** quando $\beta_5 = brkdwn$ significa que há uma descontinuidade da disponibilidade das máquinas; o tempo que uma máquina fica indisponível pode ser considerado fixo, ou ser modelado por meio de uma distribuição de probabilidade;

$\beta_6 \in (\emptyset, p_j = p, \underline{p} \leq p_j \leq \bar{p})$**:** o parâmetro β_6 define um intervalo de duração para os *jobs*; o valor $\beta_6 = (p_j = p)$ indica que todos os *jobs* possuem a mesma duração; um valor de $\beta_6 = (\underline{p} \leq p_j \leq \bar{p})$ indica que a duração dos *jobs* está entre o intervalo $[\underline{p}, \bar{p}]$;

$\beta_7 \in (\emptyset, block)$**:** descreve se o sistema pode parar devido a uma limitação de filas; pode acontecer em *flow shop*; quando um *job j* termina seu processamento em uma máquina i, mas a fila da máquina $i + 1$ está cheia, então j não pode sair de i fazendo com que os *jobs* a serem processados fiquem bloqueados; o parâmetro β_7, quando atribuído *block*, permite o bloqueio de *jobs*; caso contrário, a fila dos recursos devem ser aumentada.

$\beta_8 \in (\emptyset, no - wait)$**:** este parâmetro descreve se são permitidas esperas entre duas máquinas; isso implica, caso seja necessário, no adiamento da execução de um *job* para que ele possa ser executado sem a necessidade de fila; caso $\beta_8 = no - wait$, nenhum espaço de armazenamento entre as máquinas será definido, exigindo a transferência imediata de *jobs* entre as máquinas;

3.2.3 Critérios de Otimização (γ)

A otimização do trabalho é descrita por meio do campo γ. Este parâmetro, geralmente, é definido por funções, as quais dependem de alguns atributos dos *jobs*. Segue uma apresentação dos atributos mais utilizados:

- **completion time (C_j):** representa o momento em que o *job* j deixa o sistema;
- **unit penalty (I_j):** verifica o estado do prazo de entrega de certo *job*; I_j vale 1 quando o *job* está atrasado e 0 quando não há atraso:

$$I_j = \begin{cases} 1 & \text{se } C_j > d_j; \\ 0 & \text{caso contrário.} \end{cases} \tag{3.1}$$

- **lateness (L_j):** utiliza o prazo prometido de entrega para calcular o atraso relativo de cada *job* j; assume valor positivo, quando j está atrasado, e negativo, quando está adiantado:

$$L_j = C_j - d_j \tag{3.2}$$

- **tardiness (T_j):** refere-se ao atraso absoluto da execução do trabalho; não assume valores negativos, enfatizando somente o atraso de um certo *job* j:

$$T_j = max(C_j - d_j, 0) = \max(L_j, 0) \tag{3.3}$$

- **earliness (E_j):** refere-se ao término do trabalho em um tempo menor do que o previsto (*due date*):

$$E_j = \min(0, -L_j) \tag{3.4}$$

- **waiting time ($W_{i,j}$):** tempo precedente à i-ésima atividade do caso j, ou seja, o tempo que o *job* j deve esperar antes de começar a ser executado na i-ésima atividade; considerando uma atividade i e uma politica FIFO, o cálculo do tempo de espera em uma fila é dado pela Equação 3.5; geralmente, um valor aproximado para o tempo de processamento $p'_{i,j}$ é utilizado:

$$W_{i,j} = \sum_{i=1}^{j-1} p_{i,j} \tag{3.5}$$

As funções objetivo são consideradas critérios de otimização e são classificadas em duas famílias: *mini-max* e *minisum*. A primeira (Equação 3.6) refere-se à minimização de um valor máximo k de um conjunto de funções, e o segundo (Equação 3.7) à minimização da soma das funções [39]:

$$\int_{max}(K) := max\left\{\int_{j}(K_i)|j = 1, ..., n\right\} \tag{3.6}$$

$$\sum\int_{j}(K) := \sum_{j=1}^{n}\int_{j}(K_j) \tag{3.7}$$

Com base nesses conceitos, a seguir, listamos os possíveis valores de γ:

- **makesapan(C_{max}):** o *makespan* é equivalente ao tempo de finalização do último *job* a deixar o sistema; é definido como $C_{max} = max(C_1, C_2, ..., C_n)$; situações onde o número de casos é grande e o valor do *makespan* é baixo, geralmente representam uma alta utilização das máquinas;
- **maximum lateness (L_{max}):** métrica utilizada para medir a maior violação de prazo ocorrida no sistema; denotada por $L_{max} = max(L_1, L_2, ..., L_n)$;
- **total weighted completion time ($\Sigma\, w_j C_j$):** também referenciada como *weighted flow* time, calcula a soma ponderada dos tempos de finalização de todos os *jobs*;
- **discounted total weighted completion time ($\Sigma\, w_j(1-\exp^{-rC_j})$):** é uma métrica mais geral que a anterior, onde o custo é descontado de acordo com um fator $\{r \in \mathbb{R} : 0 < r < 1\}$ para cada unidade de tempo.
- **total weighted tardiness ($\sum w_j\, T_j$):** soma ponderada do atraso de cada *job*;
- **weighted number of tardy jobs ($\sum w_j I_j$):** nesta métrica, a importância (peso) de cada *job* é utilizada para ponderar a soma do número de casos atrasados;
- **total weighted earliness ($\sum w_j E_j$):** métrica utilizada para calcular a quantidade de prazo excedente que foi definida aos *jobs*; geralmente utilizada para adequar o *due date* d_j dos *jobs*.

De posse da descrição da notação de Graham *et al.* é possível classificar problemas de escalonamento. Na sequência, alguns exemplos da utilização desta notação são apresentados:

- $1|prec|L_{max}$**:** especifica a utilização de uma única máquina; o trabalho está sujeito à regras de precedência, geralmente definidas por meio de um grafo, e tem por objetivo minimizar o tempo máximo de atraso referente à execução dos *jobs*.

- $F5|prmtn|\sum w_jC_j$**:** denota um *flow shop* composto por cinco máquinas, ou seja, cinco máquinas em série; pode ocorrer preempção de trabalho; o objetivo é minimizar a soma ponderada dos tempos de execução.

- $J3|block|C_max$**:** representa um *job shop*, composto por três máquinas; permite a paralisação de *jobs* devido à saturação de filas; o objetivo é minimizar o tempo máximo de finalização dos *jobs*.

3.2.4 Classificação do Escalonamento

Dependendo da natureza da chegada dos trabalhos aos recursos, segundo Brucker [8], o escalonamento pode ser classificado em: *estático* ou *dinâmico*.

- **estático:** o escalonamento estático assume o conhecimento prévio do número de *jobs* e de suas respectivas características; essas informações são passadas ao escalonador que irá determinar a ordem e o local de execução de cada *job*, ou seja, o escalonador organiza os *jobs* uma única vez; apesar de ser aplicável somente a problemas bem estabelecidos, o escalonamento estático apresenta algumas vantagens, a citar: baixo custo computacional, não exigindo reescalonamento em tempo de execução, capacidade de realizar predições precisas (úteis em aplicações críticas de segurança).

- **dinâmico:** em contrapartida, escalonadores dinâmicos são projetados para lidar com imprevisibilidade, não tendo conhecimento do número nem do momento em que os casos irão chegar ao sistema; a adaptação do sistema ao ambiente deve ser constante, caso contrário, os resultados sofreriam uma deterioração a cada *job* que entrasse ou saísse do sistema; desse modo, a cada novo caso uma reorganização do escalonamento inicial se faz necessário (*rescheduling*), exigindo um bom gerenciamento para que o reescalonamento não se torne demasiadamente custoso.

Num contexto mais amplo, Pinedo [31] discute as incertezas intrínsecas a problemas reais. Por exemplo, na maioria das situações reais não se pode prever, ao menos com exatidão, quando uma máquina vai apresentar problemas e acarretar perdas no desempenho do processo. Em alguns casos, os dados disponíveis não são suficientes para modelar deterministicamente o comportamento do sistema, exigindo representações mais flexíveis dos dados. Assim, para Pinedo, pode-se também classificar um problema de escalonamento como *determinístico* ou *estocástico*.

- **modelo determinístico:** modelo no qual o estado dos *jobs* e o comportamento dos recursos são previamente conhecidos; não possuem a capacidade de expressar

incertezas e nem sempre representam de maneira correta a realidade; a solução de um problema de escalonamento é determinada apenas utilizando as condições iniciais;

- **modelo estocástico:** descreve as variáveis por meio de suas propriedades estatísticas; atributos como tempo de processamento, tempos de chegada e desempenho de recursos são representados por funções de distribuição de probabilidade; para estabelecer tais propriedades estatísticas, pode-se repetir medidas e derivar as propriedades requeridas, ou utilizar observações similares que foram realizadas no passado.

3.3 Diferenças entre Wokflow e Escalonamento

Apesar de muitos conceitos de *workflow* e escalonamento compartilharem das mesmas ideias básicas, dificuldades e incoerências são encontradas quando se deseja mapear um problema de *workflow* para a área de escalonamento. Um resumo das diferenças que causam tais dificuldades levantadas em [41], são apresentadas a seguir:

Incertezas Quanto ao Tempo de Execução das Tarefas

- **workflow:** incertezas quanto ao tempo de execução são inerentes aos recursos modelados; em um sistema de avaliação de artigos, por exemplo, o tempo de avaliação depende de qual artigo está sendo avaliado e de quem está avaliando;
- **escalonamento:** a maioria das pesquisas é focada em ambientes onde o tempo de execução das tarefas é exato e conhecido previamente, como, por exemplo, em processos de linha de montagem ou chão de fábrica, onde os tempos de execução de cada pequena etapa são conhecidos.

Incertezas Quanto às Rotas dos Casos

- **workflow:** a definição de processo modela os diferentes caminhos que uma instância pode tomar, muitas vezes sendo impossível determinar previamente a sua rota;
- **escalonamento:** o tratamento de rotas não determinísticas não foi encontrado na literatura.

Escolha da Função Objetivo

- **workflow:** diversas métricas de avaliação são utilizadas para avaliar diferentes requisitos em sistemas de *workflow*, como funções que calculam a porcentagem de casos atrasados, o atraso médio por caso atrasado; além disso, combinações de funções objetivo são utilizadas para avaliar o desempenho de cenários específicos de *workflow*;
- **escalonamento:** o escopo de funções objetivo nesta área é restrito; a maioria das pesquisas procura minimizar apenas o tempo de finalização dos jobs, ou seja, otimizar a função *flowtime*.

Preempção de Atividades

- **workflow:** o tratamento de atividades preemptivas e não preemptivas, geralmente é definido no processo de negócio de um *workflow*, a modelagem de dois tipos de atividades advém da necessidade de representar atividades urgentes (como uma reunião para tomada de decisão) e rotineiras (como a escrita de um documento, que pode ser pausada e reiniciada diversas vezes);
- **escalonamento:** a maioria das pesquisas avaliadas concentram-se em ambientes binarizados, onde todos os *jobs* são preemptivos, ou todos não o são.

Natureza Dinâmica

- **workflow:** sistemas de *workflow* são naturalmente dinâmicos, pois casos chegam e são concluídos a todo momento; o não tratamento do dinamismo do sistema pode causar a deterioração dos resultados, tornando o sistema pouco útil;
- **escalonamento:** grande parte das pesquisas são focadas em situações estáticas onde todos os *jobs* são conhecidos previamente, e uma vez gerada a ordenação, esta não muda; pesquisas que lidam com sistemas dinâmicos, decompõem um problema dinâmico em diversos problemas estáticos.

Mapeamento entre Recursos e Atividades

- **workflow:** o mapeamento entre recursos (ou máquinas em escalonamento) e atividades (passo de execução de um *job* em escalonamento) é $m : n$, ou seja, um recurso pode ser executado por meio de n atividades e uma atividade pode ser executada por m recursos;
- **escalonamento:** o mapeamento entre recursos é efetuado de forma $1 : 1$ ou $1 : n$, ou seja, um passo de execução de um *job* geralmente é feito apenas por

uma máquina, ou existem bancos de máquinas paralelas ($\alpha|\ |\ $), onde os *jobs* devem ser executados uma vez em cada um desses bancos, em qualquer de suas máquinas.

3.4 Considerações Finais

A padronização, ordenação e a formalização das atividades que compõem os processos contribuem para uma maior eficiência e produtividade nas empresas. Por esse motivo, a área de sistemas de *workflow* encontra-se em evidência atualmente. Esse conceito é cada vez mais valorizado no atual mercado competitivo, uma vez que o sistema de gerenciamento de *workflow* oferece apoio à decisões importantes nas organizações.

A busca constante por métodos que possam auxiliar na gerência e na otimização dos recursos aproxima a área de escalonamento a de *workflow*. Nesse sentido, novas formas de adequar e aplicar técnicas de escalonamento em problemas de *workflow* mostram-se promissora.

Capítulo 4

Proposta para Escalonamento Paralelo em *Workflow*

Atualmente, o tempo é cada vez mais escasso e a necessidade de respostas rápidas passou a ganhar maior importância. Apresentar respostas claras e consistentes para os clientes no menor tempo possível permite que eles tomem suas decisões rapidamente, reduzindo custos e contribuindo para um melhor gerenciamento de processos. Além disso, uma estrutura que apresente rapidez e precisão permite que um maior número de clientes possa ser atendido em num menor intervalo de tempo.

A gerência de execução de processos está diretamente ligada a sistemas de gerenciamento de *workflow*. Desse modo, aumentar a eficiência em sistemas de *workflow* implica em aumentar a qualidade no gerenciamento de recursos e tempo. Porém, aumentar a eficiência de sistemas de *workflow* não é uma tarefa trivial. Contribuem para o aumento da complexidade do problema as dificuldades em prever quanto tempo será gasto na execução de uma atividade, a diversidade de caminhos que uma tarefa pode tomar em um sistema *workflow*, a presença de dependências entre etapas (como a causada por ANDs) e as diferentes políticas de escalonamento de tarefas.

4.1 Caracterização do Problema

A modelagem de um sistema complexo de *workflow* representa as relações entre as atividades atuantes de um processo de negócio. Estas relações definem a ordem em que as atividades devem ser executadas para que o objetivo principal, definido pelo processo de negócio, seja atingido. Computacionalmente essas relações representadas por meio de um grafo dirigido. Avaliando as ligações definidas pelo grafo, quatro estruturas de roteamento podem ser identificadas: roteamento sequencial; roteamento paralelo; roteamento condicional e roteamento iterativo.

Um *workflow* complexo é formado por diversas estruturas de roteamento onde uma estrutura de roteamento pode ser composta por outras estruturas, como um LOOP dentro de um AND. Geralmente a classificação é feita de acordo com o objetivo a ser atingido. Por exemplo, suponha que dentro de uma estrutura paralela há uma estrutura de roteamento iterativa, neste caso, pode ser mais conveniente considerar apenas a estrutura paralela quando existe uma política de escalonamento específica para estruturas paralelas.

A complexidade de um sistema de *workflow* é ampliada devido às incertezas inerentes ao problema, como o não conhecimento do número de instâncias que serão executadas e a incerteza quanto ao tempo de execução de cada uma das instâncias em cada uma das atividades processantes.

Em um sistema ideal simples, cada atividade executa uma única atividade por vez e trabalhos nunca são acumulados, ou seja, não existem filas de espera. Porém, ignorar a execução paralela ou assumir a inexistência de filas não reflete a realidade dos modelos atuais de *workflow*. Diante disto, aumentar a eficiência de um sistema de *workflow* exige a elaboração de métodos eficientes para a escolha da instância de trabalho que deve ser executada. Como exemplo, considere uma sequência simples composta por uma atividade processante e duas instâncias a serem processadas, conforme mostrado na Figura 4.1. Suponha que a 1ª e a 2ª instância necessitem, respectivamente, de 10 e 20 unidades de tempo para serem executadas. Caso seja mantida uma política FIFO, a situação descrita pela Figura 4.1(a) terá um tempo médio de processamento de 40 unidades de tempo e a situação descrita pela Figura 4.1(b) terá um tempo médio de processamento de 50 unidades de tempo. Estas duas situações mostram que utilizar políticas de escalonamento eficientes é essencial para aumentar o desempenho de um sistema de workflow.

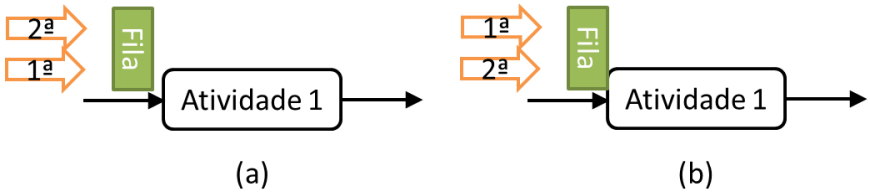


Figura 4.1: Exemplo de uma estrutura de roteamento sequencial composta por apenas uma atividade; a) assume-se a existência de uma fila onde a 1^a instância precede a 2^a instância; b) assume-se a existência de uma fila onde a 2^a instância precede a 1^a instância.

Apesar da ineficiência de algumas políticas de escalonamento, a maioria dos sistemas de *workflow* atuais utiliza a política FIFO para atribuir trabalho aos recursos, independente da estrutura de roteamento. Ocasionalmente, a política de escalonamento SIRO também é utilizada, porém sem muita contribuição na eficiência. O uso dessas politicas é justificado por sua simplicidade de utilização [42]. Surge então o questionamento sobre a

real eficiência desse tipo de abordagem nesses sistemas.

O trabalho desenvolvido por Tramontina [40] analisa a aplicação de diversas políticas de escalonamento em cada uma das estruturas de roteamento, identificando quando qual política é mais eficiente que outra. Devido à existência de algumas particularidades na estrutura de roteamento paralela (AND), utilizar políticas genéricas, ou seja, que podem ser aplicadas a qualquer outra estrutura de roteamento, se mostrou ineficiente.

A adoção de técnicas específicas de escalonamento para as estruturas do tipo AND, de modo que utilizem informações contidas somente nesse tipo de estrutura, pode ser vista como uma alternativa relevante.

4.2 Sincronismo em Estruturas Paralelas (AND)

Visualmente um AND é representado por uma bifurcação de uma linha principal de processamento em duas ou mais linhas (canais). Em cada um dos canais existem diversas atividades, sendo que uma atividade pode ser mais rápida ou mais lenta que outra. A diferença de tempos de execução também é observada em outras estruturas de roteamento, porém no caso do AND essa situação pode ser mais problemática.

Analisando o comportamento das estruturas do tipo AND nota-se que a formação de fila em uma das atividades, pode resultar na falta de sincronia entre os canais. Para ilustrar esse problema considere a Figura 4.2.

As Figuras 4.2(a) e 4.2(b) representam um mesmo AND em dois momentos distintos. No tempo 100 a instância 2 acaba de entrar no AND, e, consequentemente, foi dividida em duas partes, uma enviada ao canal 1 e a outra ao canal 2 (Figura 4.2(a)). Após 500 unidades de tempo (Figura 4.2(b)), nota-se que as partes de um mesmo caso (instância 1) estão se distanciando uma da outra, ou seja, uma das partes será concluída antes da outra. Essa diferença nos tempos de conclusão dentro de um AND caracteriza uma falta de sincronismo. Em políticas assíncronas é natura ocorrer o cruzamento de partes de instâncias distintas, como o caso da instância 1 e 2 mostrado na Figura 4.2.

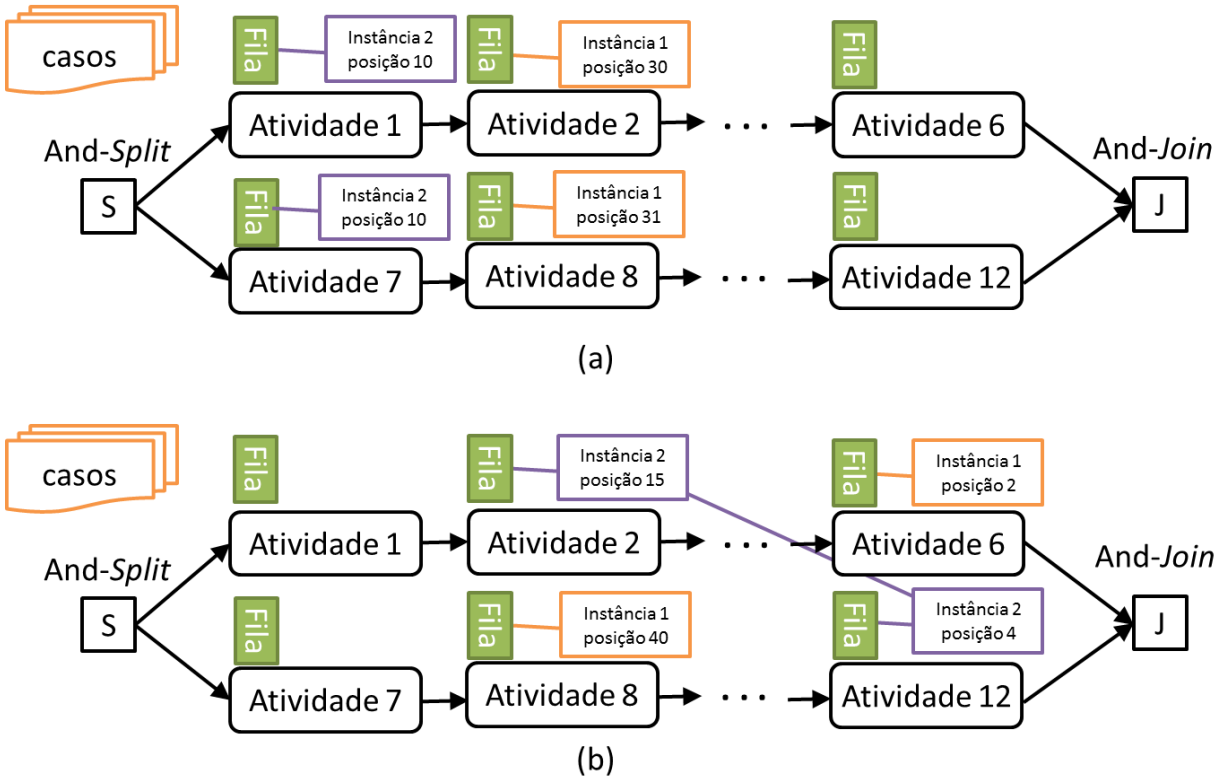


Figura 4.2: Representação de um AND de seis atividades em cada canal; a) instante de tempo 100, onde a instância 2 acaba de entrar no AND e a instância 1 já esta mais a frente no AND; b) instante de tempo 500, onde as partes de uma mesma instância se distanciam e se cruzam dentro do AND.

O sincronismo, ou a falta dele, também pode ser representado por meio de posições dentro de uma fila em um AND. Partes de um mesmo caso que possui posição próximas são considerados síncronos (Figura 4.3(a)) e partes distantes são considerados assíncronos (Figura 4.3(b)).

Algumas políticas de escalonamento podem ser consideradas naturalmente síncronas pois mantém uma distância pequena entre as partes de um mesmo caso. Nessas políticas o cruzamento de partes de casos distintos não ocorre. Uma política naturalmente síncrona muito utilizada em sistemas de *workflow* é a FIFO. Apesar do sincronismo da FIFO, ela não faz uso de correlação entre as partes de um mesmo caso caindo em situações desfavoráveis e ineficientes. Por exemplo, considere a Figura 4.4.

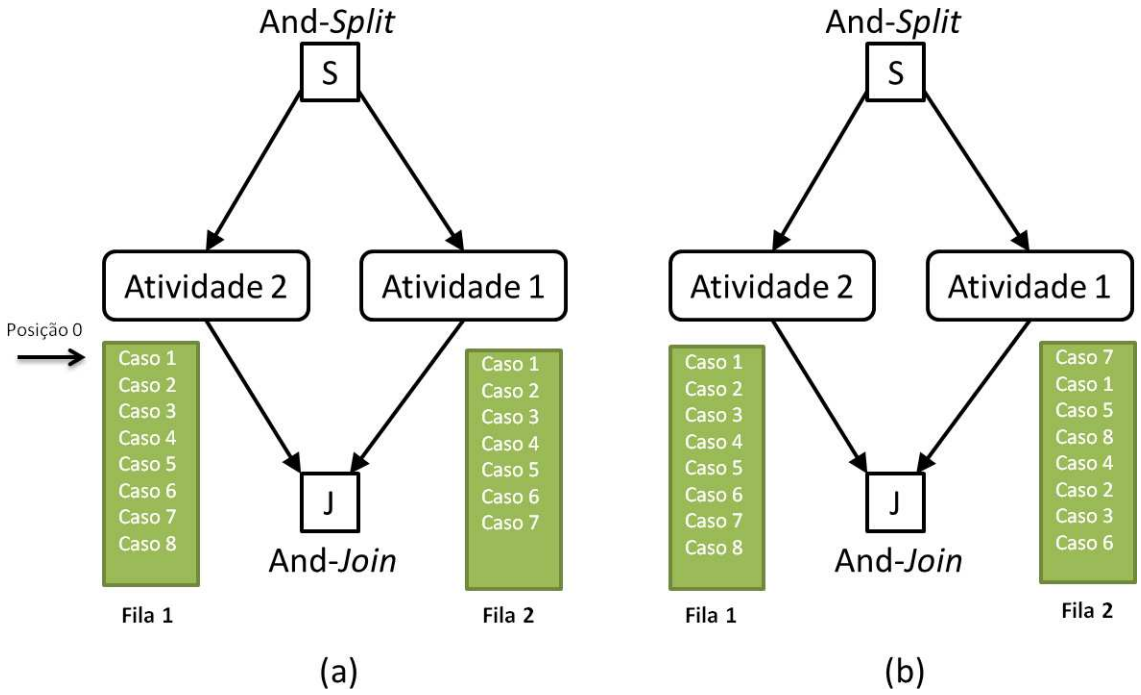
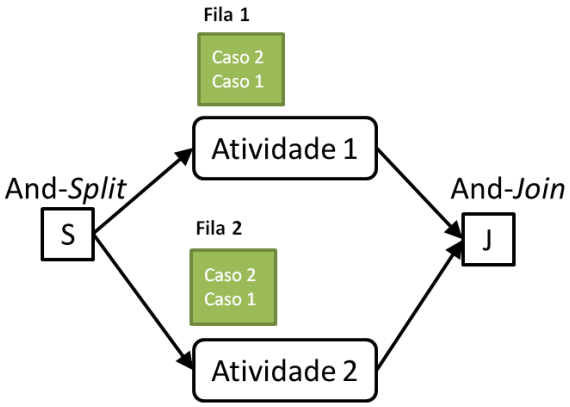


Figura 4.3: Representação de sincronismo entre filas dentro de um AND; a) instâncias sincronizadas; b) instâncias assincronas.



	Atividade 1	Atividade 2
$Caso1$	$p_{1,1} = 100$	$p_{1,2} = 98$
$Caso2$	$p_{2,1} = 1$	$p_{2,2} = 5$

Figura 4.4: Representação de um AND e duas instâncias sendo executadas por meio de uma política de escalonamento FIFO.

Na situação descrita pela Figura 4.4 é fácil perceber que manter a ordem de chegada das tarefas não é uma boa escolha, pois a instância 1 consome muito tempo de processamento, e, caso seja mantida esta ordem, o tempo médio de processamento será alto. Uma possível solução é utilizar SPT para ordenar os casos, reduzindo o tempo médio de processamento

de 100,5 (tempo de exec. Caso 1 + tempo de exec. Caso 2 = 100 + 101) para 51,5 unidades de tempo (tempo de exec. Caso 2 + tempo de exec. Caso 1 = 1 + 101). Porém SPT não é síncrono e a falta de sincronismo pode levar a SPT se comportar pior que FIFO em algumas situações. Um exemplo de situações que devem ser eviadas é representado pela analise das Tabelas: 4.1, 4.2 e 4.3; e pelo cenário apresentado pela Figura 4.5.

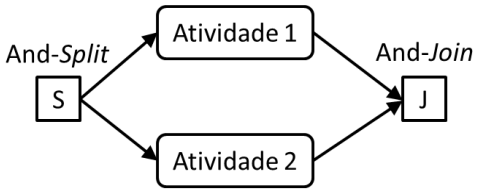


Figura 4.5: Cenário paralelo simples.

Caso j \ Atividade i	$i = 1$	$i = 2$
$j = 1$	$p_{1,1} = 1$	$p_{1,2} = 3$
$j = 2$	$p_{2,1} = 2$	$p_{2,2} = 2$
$j = 3$	$p_{3,1} = 3$	$p_{3,2} = 1$

Tabela 4.1: Tempos de execução dos casos j em cada uma das atividades i ($p_{\mathrm{i,j}}$).

Caso \ Atividade	*Atividade*1	*Atividade*2	$AND - Join$
Caso 1	$C_{1,1} = 1$	$C_{1,2} = 3$	$max(1, 3) = 3$
Caso 2	$C_{2,1} = 3$	$C_{2,2} = 5$	$max(3, 5) = 5$
Caso 3	$C_{3,1} = 6$	$C_{3,2} = 6$	$max(6, 6) = 6$
			Tempo Médio de Exec. = 4.667

Tabela 4.2: Tempo de conclusão FIFO.

Caso \ Atividade	*Atividade*1	*Atividade*2	$AND - Join$
Caso 1	$C_{1,1} = 1$	$C_{3,2} = 6$	$max(1, 6) = 6$
Caso 2	$C_{2,1} = 3$	$C_{2,2} = 3$	$max(3, 3) = 3$
Caso 3	$C_{3,1} = 6$	$C_{1,2} = 1$	$max(6, 1) = 6$
			Tempo Médio de Exec. = 5

Tabela 4.3: Tempo de conclusão SPT.

Os valores contidos nas colunas 2 e 3 ($C_{i,j}$) das Tabelas 4.2 e **??** representam os tempos de conclusão de cada caso i na atividade j. O valor da terceira coluna representa o momento que todas as partes de um caso chega ao AND-*Join*, permitindo que o caso seja dado como concluído.

O problema ocorrido na política SPT (Tabela 4.3) se deve ao tempo de espera. Para que uma instância seja concluída em um AND todas as partes de uma mesma instância devem chegar ao AND-*Join*. Caso uma parte de uma instância seja concluída ela ficará esperando até que as outras partes também terminem, porém esse tempo de espera pode ser considerado como desperdício de recurso. Para aproveitar o benefício da política SPT e evitar situações semelhantes as mostrada na Tabelas 4.3 é necessário considerar o sincronismo.

Para analisar o sincronismo, primeiro é necessário definir uma métrica de distância. Para o cenário descrito pela Figura 4.5 e pelas Tabelas: 4.1 e 4.3, onde existe apenas uma atividade por canal do AND, uma possível métrica é a diferença entre as posições nas filas 1 e 2 para as partes de um mesmo caso. Calculando as distâncias para o escalonamento feito pela política SPT, obtêm-se os resultados apresentados na Tabela 4.4.

Ordem Ex: Ativ. 1	Ordem Ex: Ativ. 2	Dist.	$AND - Join$
$Caso1 - C_{1,1} = 1$	$Caso3 - C_{3,2} = 1$	$abs(1-3) = 2$	$max(C_{1,1}, C_{1,2}) = 6$
$Caso2 - C_{2,1} = 3$	$Caso2 - C_{2,2} = 3$	$abs(2-2) = 0$	$max(C_{2,1}, C_{2,2}) = 3$
$Caso3 - C_{3,1} = 6$	$Caso1 - C_{1,2} = 6$	$abs(3-1) = 2$	$max(C_{3,1}, C_{3,2}) = 6$
			Tempo Médio de Exec. = 5

Tabela 4.4: Tempo de espera que uma parte precisa esperar até a outra chegar ao AND-*Join* para SPT: sem considerar a correlação entre as partes de um caso.

A coluna 1 da Tabela 4.4 representa a ordem de execução de cada um dos casos na atividade 1 e seu respectivo tempo de conclusão caso seja mantida esta ordem. A coluna 2 representa a mesma situação da coluna 1, porém para a atividade 2. A terceira coluna representa os valores absolutos das diferenças entre: posição da fila de uma parte de um caso na atividade 1 e sua outra metade na atividade 2. Por fim, a última coluna representa os tempos de conclusão de cada caso após saírem do AND-*Join*.

Analisando a Tabelas 4.4 é possível verificar que as maiores diferenças se encontram nas posições 1 e 3. Esses valores representam situações de falta de sincronismo. Para reduzir as diferenças, a parte do caso que se encontra na maior posição da fila dever ser reescalonada para uma posição menor (será executada antes). Realizando essa alteração obtêm-se o resultado apresentado na Tabela 4.5.

Ordem Ex: Ativ. 1	Ordem Ex: Ativ. 2	Dist.	$AND - Join$
$Caso3 - C_{3,1} = 3$	$Caso3 - C_{3,2} = 1$	$abs(1-1) = 0$	$max(C_{1,1}, C_{1,2}) = 3$
$Caso2 - C_{2,1} = 5$	$Caso2 - C_{2,2} = 3$	$abs(2-2) = 0$	$max(C_{2,1}, C_{2,2}) = 5$
$Caso1 - C_{1,1} = 6$	$Caso1 - C_{1,2} = 6$	$abs(3-3) = 0$	$max(C_{3,1}, C_{3,2}) = 6$
			Tempo Médio de Exec. = 4,667

Tabela 4.5: Tempo de espera que uma parte precisa esperar até a outra chegar ao AND-*Join* para SPT: considerando a correlação entre as partes de um caso.

Analisando a Tabela 4.5 é possível observar que a distância relativa entre as partes dos casos caiu. Esse melhoria no sincronismo resultou na redução do tempo médio de processamento de 5 para 4,667, quando comparado a política SPT sem sincronia. Com essa modificação é possível manter os bons resultados da política SPT evitando situações adversas, como mostrado na Tabela 4.3. Estruturas paralelas maiores exigem diferentes métricas de medidas, pois além da distância relativa a fila existe à distância entre as atividades.

Decidir qual métrica utilizar para calcular a distância entre as partes de um caso e utilizá-las de modo adequado para alterar a prioridade dos casos é fundamental para melhorar a eficiência de políticas de escalonamento em ANDs.

Em algumas situações, a falta de sincronismo é causada pela disposição das atividades dentro do AND (Figura 4.6). Nesses casos temos um problema de sincronismo estrutural. Mesmo que todas as atividades possuam o mesmo comportamento de execução, a tendência é que as atividades do canal 1 sejam concluídas após as atividades do canal 2.

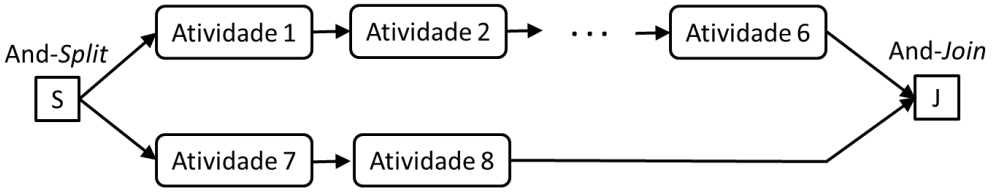


Figura 4.6: Exemplo de um dessincronismo estrutural em um AND

Apesar das situações mais extremas (Figura 4.6), existem situações que permitem uma correção de possíveis falhas no sincronismo caudas pela diferença de tempo de execução das atividades. Por exemplo, considere a situação representada pela Figura 4.7.

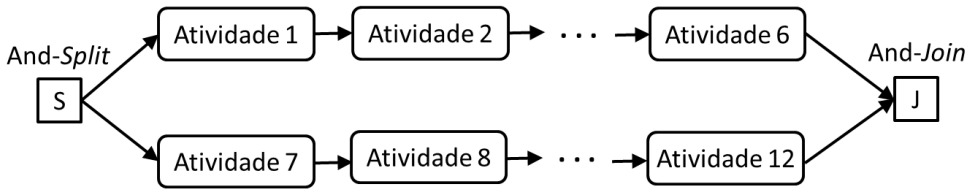


Figura 4.7: Exemplo de um AND com canais simétricos, seis atividades em cada canal.

Na Figura 4.7 temos seis atividades em cada canal, assim, caso aconteça uma falta de sincronismo no início dos dois canais, esse, devido à existência de várias filas e de políticas de escalonamento específicas pode ser resolvido mais a frente apenas alterando a prioridade de cada parte de uma mesma instância.

O sincronismo de instâncias dentro de um AND depende da correlação existente entre as partes de uma mesma instância. Utilizando tais informações é possível mensurar essa diferença (distância entre as partes) e utiliza-la para sincronizar o sistema. Para contribuir com a resolução deste problema, três políticas específicas de escalonamento para estruturas de roteamento paralelo foram propostas: *Paralel* FIFO-*DueDate*; *Synchronized*-SPT; Process Time Ratio.

4.3 *Parallel* FIFO-*DueDate*

Esta heurística tem por finalidade minimizar o número de casos atrasados de estruturas de roteamento AND. O princípio desta heurística é aumentar o sincronismo das partes de um mesma instância, mantendo uma prioridade comum para partes de um mesmo caso que possui um *due date*, semelhante à política FIFO, e priorizando pedaços com prazos mais reduzidos, semelhante ao EDD.

Dado o ambiente heterogêneo de atividades, onde diferentes atividades possuem eficiência distintas de processamento, manter o sincronismo perfeito e ao mesmo tempo evitar o atraso de casos não é uma tarefa plausível, porém reduzir a distância dos tempos entre as partes de um caso é. A medida utilizada para ponderar a distância (sincronismo) entre as partes é definida pela Equação 4.1.

$$Lag_j = d_j - DDFifo_j \tag{4.1}$$

onde:
$DDFifo = \max(DDFifo_{j,1}, DDFifo_{j,2}, ..., DDFifo_{j,l})$

O atributo $DDFifo_j$ representa o tempo que um certo caso j levaria para ser concluído, caso uma política FIFO fosse mantida e nenhum outro caso chegasse ao sistema.

Desse modo, a diferença entre o prazo para o termino (d_j) e a previsão FIFO ($DDFifo_j$) fornecem um meio para avaliar como anda o processamento de todas as partes de uma mesma instância.

Definir o momento em que certa instância se torna mais propensa a se tornar atrasada não é um processo discreto, pois durante uma execução de um sistema de *workflow* muitas instâncias possuem prazos e Lag_j semelhantes ao mesmo tempo que mantêm posições em filas bem divergentes. Para suavizar a classificação de instâncias atrasadas, ou não atrasadas, uma modelagem sigmoidal (Figura 4.8) foi utilizada (Equação 4.2). Esta função é utilizada para representar a transição entre os dois estados: o primeiro representando instâncias não atrasadas, ou seja, com baixa prioridade ("parte baixa da sigmoide"); e o segundo representando os casos atrasados, com mais prioridades ("parte alta da sigmoide").

$$S(x) = \frac{1}{1 + \exp^{-x}} \tag{4.2}$$

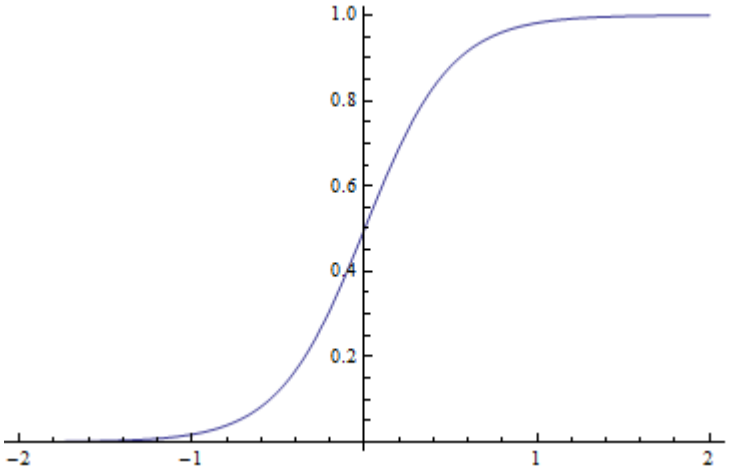


Figura 4.8: Representação gráfica de uma sigmoide (Equação 4.2)

Ajustando a sigmoide aos parâmetros de distância e tempo mínimo de execução necessário para a conclusão de certa instância obtemos a seguinte Equação 4.3.

$$DD_{\text{int}} = (1 + Lag_j)\left[\frac{1 - \frac{max(\overline{p}'_j)}{(1+Lag_j)}}{1 + e^{-\epsilon.(Lag_j - \varrho)}}\right] + max(\overline{p}'_j)) \tag{4.3}$$

onde:

ϵ representa a suavidade da transição, experimentalmente definida em 0.01;

ϱ definada como a mediana dos valores de $DLag_j$ dos casos de uma mesma atividade.

Nesta heurística, uma instância que tenha um $DDFifo_j$ compatível com seu prazo, ou seja, que não vai atrasar se ele mantiver o ritmo de execução, nenhuma intervenção é efetuada. Para os casos sem sincronismo (alto valor de $DDFifo_j$), em que os prazos não serão cumpridos casos se mantenham no mesmo ritmo de execução, a prioridade das partes mais atrasadas são aumentadas. A definição da prioridade dos casos é dada por DD_{int} (eixo das ordenadas), onde valores baixos representam casos com baixa prioridade e altos casos com alta prioridade.

4.4 *Synchronized*-SPT

O critério de otimização da *Synchronized*-SPT é minimizar o tempo de processamento médio. Esta heurística baseia-se no bom desempenho apresentado por SPT em cenários simples de escalonamento, quando o tempo médio de processamento é a métrica de avaliação [10]. Um AND é caracterizado pela existência de canais, geralmente formados por atividades em sequência simples, o que pode favorecer a aplicação desta técnica em ANDs. Porém, a simples aplicação de SPT em sistemas paralelos nem sempre produz bons resultados devido ao problema de sincronismo.

Para solucionar o problema de sincronismo da política SPT três etapas são executadas: 1^a) a política SPT é aplicada em cada um dos canais do AND; 2^a) uma métrica de distância para o sincronismo é calculada; 3^a) "falsos" valores de tempo de processamento esperado $p'_{i,j}$ são introduzidos ao escalonador para que ele dê maior prioridade às partes mais atrasadas de uma instância.

Porém, aumentar a prioridade de uma ou mais partes não resulta necessariamente em uma redução no tempo de processamento. Tomemos o caso em que uma parte de uma instância está atrasada e sempre se encontra na primeira posição da fila (sempre é a primeira a ser executada). Neste caso, aumentar a prioridade não irá resultar na diminuição do tempo de processamento, pois nenhum tempo é desperdiçado esperando em filas. Desse modo, antes de aumentar a prioridade de certa instância é necessário ter uma previsão de quanto tempo essa instância irá gastar esperando em filas e assim redefinir a prioridade proporcionalmente ao tempo de espera.

Pode-se concluir que o ganho no tempo de processamento é condicionado ao tempo que certa instância irá gastar esperando em filas. Por exemplo, considere uma situação onde um caso $j = 4$ está contido em uma fila com três outros casos a sua frente ($Fila = \{1, 2 \text{ e } 3\}$). No momento em que o recurso fica livre, o escalonador terá que decidir qual dos casos tem a maior prioridade. O ganho máximo que o escalonador pode trazer ao tempo de processamento do caso 4 é atribuí-lo imediatamente, trazendo um "ganho" referente aos tempos de processamento dos casos que estão a sua frente.

Desse modo, para cada instância j do sistema, podemos decompor o tempo total de

execução de um caso S_j em duas partes: tempo gasto processando (p'_j) e tempo gasto em espera em filas (w_j), como definido na Equação 4.4.

$$w_j = S_j - p'_j \tag{4.4}$$

O tempo mínimo necessário para executar uma instância (p'_j) é obtido avaliando o comportamento de execução de cada uma das atividades. Desse modo, para calcular o tempo de espera é necessário obter uma previsão do tempo total de processamento de uma instância dentro de um cenário específico.

O cálculo do tempo total de processamento está relacionado com a política de escalonamento adotada. Dependendo da política, realizar os cálculos necessários para obter o valor do tempo total se torna inviável. Para reduzir o processamento necessário, o tempo total de processamento foi calculado considerando uma política FIFO de execução, igualmente a utilizada na política *Paralel* FIFO-*DueDate*.

Tendo em mãos as previsões de tempo que cada instância irá gastar, basta utilizar essa informação para alterar proporcionalmente a prioridade delas. A política SPT utiliza os tempos de processamento das atividades para decidir qual instância deve ser executada, sendo priorizadas as instâncias com menor tempo de execução. Desse modo, a manutenção do sincronismo é efetivada manipulando os tempos de processamento visíveis ao escalonador, reduzindo o valor quando a parte da instância esta mais atrasada e mantendo o valor para a parte que não está atrasada.

A proporção de redução dos tempos de processamentos é definida pelo coeficiente de sincronismo ϑ. Esse coeficiente é utilizado para multiplicar os valores esperados de tempo de processamento $p'_{i,j}$ para formar um novo valor $p''_{i,j}$. O atributo $p''_{i,j}$ é um valor temporário, pois a cada novo escalonamento um novo valor para $p''_{i,j}$ é calculado com base na Equação 4.4.

$$\vartheta_{\mathrm{j}} = \begin{cases} EW & \text{se } \; p'_{\mathrm{j}} > w_{\mathrm{j}}; \\ (1 - EW) & \text{caso contrário.} \end{cases} \tag{4.5}$$

onde:

$$EW = p'_{\mathrm{j}} \,/\, w_{\mathrm{j}}$$

Dessa forma, uma redução proporcional ao tempo que a instância j irá gastar em filas é imposta a parte de uma mesma instância que está atrasada. Esta redução fará com que a prioridade da parte atrasada aumente quando essa for ordenada pelo menor tempo de processamento (SPT).

4.5 Process Time Ratio

O critério de otimização desta heurística é minimizar o tempo de processamento médio. Seu principio é manter o tempo de saída previsto das duas partes de um mesmo caso semelhante ao tempo de processamento mínimo necessário para concluir o caso. Para tal, uma relação entre o tempo médio de processamento dos casos, em um cenário previamente determinado, e a previsão de conclusão para o caso dentro do AND é calculada (Equação 4.6).

$$Pri_{\mathrm{j}} = \frac{\overline{p}'_{\mathrm{j}}}{DDFifo_{\mathrm{j}}} \tag{4.6}$$

O valor do tempo médio de processamento esperado para um caso qualquer $\overline{p}'_{\mathrm{j}}$ pode ser calculado através da Equação 5.6. O $DDFifo$ é calculado da mesma forma que foi atribuído para as heurísticas anteriores.

Desse modo, valores próximos a 1 representam situações onde o tempo de processamento do caso está próximo do mínimo necessário para a conclusão, ou seja, pouco tempo será gasto em filas. Um valor grande de Pri representa situações onde o caso necessita de muito tempo para ser processado por completo, porém possui um valor de espera em filas baixo, não necessitando um aumento de prioridade. Por fim, valores menores que 1 representam casos que necessitam de pouco tempo para serem processado, porém gastarão muito tempo em filas. Um resumo das prioridades dessa política é apresentado na Tabela 4.6.

$\overline{p}'_{\mathrm{j}}$	$DDFifo_{\mathrm{j}}$	$\overline{p}'_{\mathrm{j}}/DDFifo_{\mathrm{j}}$	Prioridade
$\uparrow$	$\uparrow$	$Pri \approx 1$	normal
$\uparrow$	$\downarrow$	$Pri \gg 1$	baixa
$\downarrow$	$\uparrow$	$Pri \ll 1$	alta
$\downarrow$	$\downarrow$	$Pri \approx 1$	normal

Tabela 4.6: Resumo das prioridades da política de escalonamento *Process Time Ratio*

4.6 Considerações Finais

A distribuição de tarefas em um sistema de *workflow* dinâmico e muitas vezes estocástico não possui uma solução simples. A análise detalhada das diferentes estruturas presentes em um sistema de *workflow* contribui para um melhor entendimento do comportamento desses diferentes fluxos de dados. A junção desse conhecimento com o desenvolvimento

de políticas de escalonamento especializadas contribui para o aumento da eficiência do problema de despacho de tarefas.

Neste capítulo, foram apresentadas três heurísticas de escalonamento voltadas a estruturas de roteamento paralelas (AND) em sistemas de *workflow*. Tais políticas utilizam os conceitos oriundos da área de escalonamento para tentar melhorar o sincronismo de casos dentro de um AND, mantendo um baixo custo computacional.

Os resultados obtidos pelas heurísticas propostas e a interpretação dos mesmos são apresentados respectivamente nos Capítulos 5 e 6.

Capítulo 5

Configurações de Simulação

Esse trabalho propõe que é possível melhorar a eficiência de estruturas de roteamento do tipo AND por meio do uso de políticas de escalonamento específicas, as quais utilizem informações contidas nos diferentes canais desta estrutura. Para tal, um conjunto de etapas foi elaborado efetivar a modelagem e a implementação do simulador dinâmico de *workflow*. Essas etapas são descritas s seguir:

1. Definição dos cenários básicos de *workflow* de modo que contenham as características desejadas;
2. Definição do comportamento das atividades, bem como dos tempos de execução de cada caso;
3. Atribuição dos prazos para cada caso do sistema;
4. Apresentação do sistema de controle de carga e gerenciamento de filas;
5. Definição das métricas utilizadas para avaliar cada aspecto dos cenários, que cobrissem situações tanto dependentes dos prazos dos casos quanto independentes;
6. A simulação de execução de processos nos cenários básicos propostos;
7. Análise dos resultados das simulações, para a efetiva avaliação do comportamento das técnicas de escalonamento propostas segundo as métricas estabelecidas;

5.1 Cenários Básicos Estudados

Cada cenário do AND possui duas estruturas principais: atividades de processamento e estruturas de roteamento. As atividades de processamento são aquelas onde os casos são

efetivamente executados. Cada caso gasta um tempo de processamento finito e maior que zero para ser concluído. O tempo de execução que cada caso irá gastar em cada um dos recursos é definido de acordo com uma distribuição de probabilidade normal de média previamente estabelecida (*modelo estocástico*). Já as estruturas de roteamento são utilizadas para direcionar os casos para a próxima atividade. Vale ressaltar que não é computado o gasto de tempo na simulação quando os casos estão passando pelas atividades de roteamento.

Outras duas estruturas de controle são encontradas no AND. A primeira é uma estrutura chamada de AND-*Split*. Esta estrutura fica localizada no inicio do AND, e é utilizada para copiar o caso no número correto de canais que o AND possui. Ou seja, assim que um caso chega a um AND, ele é copiado para cada canal do AND, sendo essas cópias executadas paralelamente. Quando a primeira parte do caso chega ao final do seu canal de processamento, esta é roteada para outra estrutura, denominada AND-*Join*. Tal estrutura é utilizada como uma espécie de *buffer* de sincronismo que fica aguardando as outras partes do caso para que ele possa ser dado como concluído.

Para avaliar o comportamento das diferentes políticas de escalonamento três configurações básicas de cenários foram organizadas, tal como apresentado a seguir:

Cenário-1 dois canais contendo duas atividades de processamento cada; este cenário tem por objetivo avaliar situações onde as metades, de um mesmo caso não podem se distanciar muito em relação ao número de atividades processantes (Figura 5.1).

Cenário-2 dois canais contendo números distintos de atividades processantes; neste cenário, um dos canais possui duas atividades de processamento e outro seis atividades; o objetivo é avaliar a discrepância entre o fluxo de processamento, ou seja, avaliar a capacidade das políticas de escalonamento manter em sincronismo os casos mesmo em um ambiente não propício, onde a profundidade de cada caso é bem discrepante (Figura 5.2).

Cenário-3 dois canais com um grande número de atividades processantes; os dois canais possuem seis atividades cada; o objetivo é avaliar o comportamento das politicas de escalonamento quando é possível identificar o distanciamento entre as duas metades de um caso e tentar corrigir um possível problema de sincronismo (Figura 5.3).

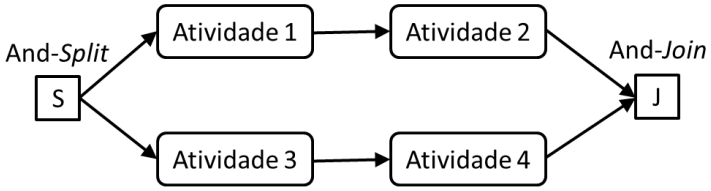


Figura 5.1: Representação gráfica do Cenário-1.

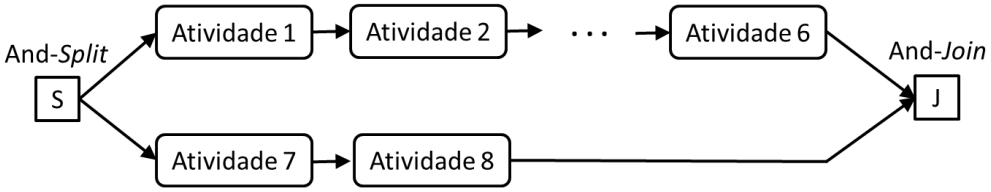


Figura 5.2: Representação gráfica do Cenário-2.

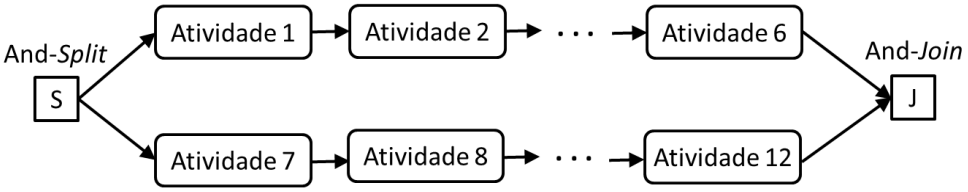


Figura 5.3: Representação gráfica do Cenário-3.

5.2 Definição do Tempo de Execução dos Casos

O comportamento de cada atividade processante é definido por meio de uma distribuição f de probabilidade normal de média μ e desvio padrão σ conhecidos:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{5.1}$$

Após a construção das distribuições, valores menores que um limite inferior ($LIM_{\min}$), ou maiores que um limite superior ($LIM_{\max}$), são descartados.

$$LIM_{\max} = 3\sigma + \mu \tag{5.2}$$

$$LIM_{\min} = \begin{cases} 1 & \text{se } (\mu - 3\sigma) \leq 0; \\ \mu - 3\sigma & \text{caso contrário.} \end{cases} \tag{5.3}$$

Estes limites têm por finalidade impedir que um tempo negativo ou um valor muito alto seja atribuído a uma atividade processante.

O tempo que cada caso irá necessitar para ser totalmente processado em uma dada atividade é definido de acordo com as distribuições de probabilidade das atividades. Desse modo, cada caso j armazena um conjunto de tempos de execução $ExT = \{p_{1,j}, p_{2,j}, \ldots, p_{M_j,j}\}$, onde $p_{i,j}$ representa o tempo de processamento de um caso j em uma atividade i, e M_j o conjunto de atividades pelas quais um caso j deve passar durante sua execução. Cada unidade de tempo refere-se a um ciclo de execução, onde ($p_{i,j} \in \mathbb{N}^+$).

Os valores de $p_{i,j}$, para cada um dos casos j, em cada uma das atividades i, é atribuído antes de iniciar a simulação. Isso permite que os mesmos tempos de execuções sejam utilizados em diferentes simulações, melhorando a comparação entre as diferentes políticas de escalonamento. Porém, esse atributo não é visível ao escalonador. O conhecimento prévio dos valores exatos dos tempos de processamento para cada um dos casos não reflete a realidade em sistemas de *workflow* (sistemas estocásticos). Desse modo, um método para inserir incertezas quanto ao tempo de execução se faz necessário.

5.2.1 Estimativa do Tempo de Execução

Os tempos de execução necessários para muitas políticas de escalonamento são representados por previsões controladas. Lawrence e Sewell [26] argumentam que é possível utilizar os valores esperados das variáveis para representar o tempo de processamento dos casos, e assim tratar o problema de forma determinística. A atribuição dos tempos de execução utilizados pelo escalonador p' segue o mesmo princípio, mas diferentemente do trabalho de Lawrence e Sewell, não opera com médias de distribuições.

O cálculo do tempo de execução utilizado pelo escalonador é efetivado por meio de uma estimativa controlada, ou seja, um erro máximo er_p, para cada previsão, é estabelecido. O sistema de previsão utiliza o tempo de processamento real p_j mais um ruído $f_{\text{ruído}}$ para calcular um novo valor para o tempo de execução $p'_{i,j}$. Assim, a inequação 5.4 é válida para toda a previsão efetuada, onde $0 \leq er_p \leq 1$:

$$er_p \geq \left| \left(\frac{p_{\text{i,j}} + f_{\text{ruído}}}{p_{\text{i,j}}} \right) - 1 \right| \tag{5.4}$$

Dependendo do problema e da quantidade de dados disponíveis, diferentes modelos de previsão podem ser utilizados. Em situações onde o tempo é um fator crítico, exige-se a elaboração de modelos mais simples e eficientes; em contrapartida, para casos onde é necessária uma maior acurácia e o tempo de resposta do previsor é menos relevante, abordagens como aprendizagem de máquina podem ser utilizadas [30]. Vale ressaltar

que, independente da metodologia adotada, ela deve possuir um erro máximo conhecido, permitindo uma melhor avaliação dos resultados [26].

Segundo Baggio *et al.* [41], a análise de diferentes situações de erro em ambientes reais revela que uma taxa de 30% de erro é uma situação passível de ocorrência. Desse modo, neste trabalho três situações de erros foram estabelecidas: 10% (mais otimista); 30% (mais próxima da realidade); e 50% (situação mais extrema).

5.3 Definição dos Prazos

Problemas reais lidam com restrições temporais, muitas delas rígidas, como as etapas necessárias para a doação de um órgão humano. Restrições temporais são basicamente representadas de duas formas: a primeira é o prazo que certo objetivo deve cumprir, denominado pela literatura de *due date*; a segunda refere-se à sincronização de esquemas complexos para permitir interação de atividades ao longo do tempo, ou simplesmente sincronização [44].

Desse modo, para avaliar quanto um caso está atrasado ou se ele foi concluído com certa antecedência, o *due date* de um caso j, ou d_j, é utilizado. Para calcular o d_j, três parâmetros são necessários: *tempo de chegada ao sistema* (r_j), *tempo total de processamento esperado para certo caso* ($\overline{p}'_{i,j}$), e *allowance factor* (A). O d_j é expresso segundo a Equação 5.5:

$$d_j = r_j + A \,.\, \overline{p}'_{i,j} \tag{5.5}$$

5.3.1 Tempo de Chegada (r_j)

O tempo de chegada r_j de cada caso depende diretamente do número de casos existentes no sistema, que por sua vez, depende das taxas de chegada e de saída dos casos. Definir uma carga de trabalho ao sistema, ou seja, controlar o número de casos existentes em um sistema de *workflow*, é denominado de *workload* [31, 25, 29].

O controle de carga utilizado neste trabalho foi o modelo proposto por Mattfeld e Bierwirth [29]. Nesse modelo, o *workload* do sistema é controlado por meio da variação do tempo de chegada entre dois casos consecutivos. Caso os tempos sejam ampliados, os casos vão chegar mais espaçados, diminuindo a carga no sistema. Porém, se o tempo for reduzido, a taxa de chegada de casos será maior, aumentando a carga do sistema.

Segundo Laguna e Marklunk [25], o comportamento das chegadas de casos em filas, em muitas situações reais, é descrito como uma distribuição exponencial com média λ. No modelo utilizado, o λ representa o tempo médio entre a chegada de dois casos consecutivos. Desse modo, para uma melhor representação do comportamento das chegadas dos casos,

os tempos entre a chegada de dois casos consecutivos foram gerados de acordo com uma distribuição exponencial de média λ.

No modelo de Mattfeld e Bierwirth [29], o λ é obtido da seguinte forma: $\lambda = \overline{p} = mU$, O m representa o número de atividades no sistema, o U a taxa de utilização desejada para as atividades e $\overline{p}$ o tempo de processamento esperado para um caso qualquer.

Para que, o cálculo do $\overline{p}$ seja efetivado, as informações de tempo de execução em cada atividade, armazenadas em cada um dos casos, são utilizadas para calcular o tempo médio com que os casos são executados.

No modelo de *workload* utilizado, o número m está associado ao número de atividades consecutivas as quais um caso deve passar obrigatoriamente antes de ser concluído, dando uma noção de profundidade de etapas. A existência de canais paralelos no AND reduz o número máximo de etapas que um caso pode passar, para o número de atividades do maior canal, e não para a soma de todas as atividades da estrutura. Por exemplo, no caso de AND que possua apenas dois canais, que possuam consecutivamente 6 e 2 atividades (Figura 5.2), a maior linha de execução é 6 (profundidade igual a 6), sendo este o valor que deve ser atribuído a m.

A variável U define a taxa de utilização do sistema, ou seja, quanto as atividades devem permanecer ativas durante a execução do sistema. Assim, atribuindo valores pequenos, estamos aumentando o espaçamento entre a chegada dos casos, diminuindo formação de filas internas e reduzindo o tempo que as atividades permanecem ativas. Caso este valor seja aumentado, um sistema mais carregado será modelado, pois os casos irão chegar a uma frequência maior, aumentando a ocorrência de filas e mantendo as atividades processantes ativas.

Por fim, de posse dos tempos de chegada entre os casos, e definindo o tempo 1 como sendo o início da execução do sistema de *workflow*, r_j pode ser definido.

5.3.2 Tempo Total de Processamento Esperado ($\overline{p}'_j$)

O tempo total de processamento esperado $\overline{p}'_j$ é calculado utilizando os tempos de execução previstos $p'_{i,j}$, armazenados em cada um dos casos. Para calcular $\overline{p}'_j$ de cada caso, um ambiente exclusivo é considerado, ou seja, o caso é executado no cenário em que será avaliado, porém sem a existência de outros casos, filas ou a possibilidade de *block* (sessão 3.2.2).

Neste trabalho, a ênfase é dada às estruturas de roteamento do tipo AND. Desse modo, o $\overline{p}'_j$, também chamado de tempo mínimo, é atribuído de modo que seja igual ao tempo de execução do canal mais lento do AND (Equação 5.6). Por exemplo, dado um AND com dois canais de execução, se em um dos canais o caso gastar 100 unidades de tempo para ser concluído e em outro gastar 50, então o tempo mínimo de execução é igual a

$\max(100, 50) - 100$,

$$\overline{p}'_{\mathrm{j}} = \max\left(\sum_{i=\underline{ch}_1}^{\overline{ch}_1} p_{i,j} \, , \sum_{i=\underline{ch}_2}^{\overline{ch}_2} p_{i,j} \, , ..., \sum_{i=\underline{ch}_{\mathrm{l}}}^{\overline{ch}_{\mathrm{l}}} p_{i,j} \right) \tag{5.6}$$

onde:
l representa o número de canais existentes no AND;
$\underline{ch}_{\mathrm{l}}$ corresponde à primeira atividade do canal l;
$\overline{ch}_{\mathrm{l}}$ corresponde à última atividade do canal l.

É importante lembrar que, diferente dos tempos de execução reais, os valores de $p'_{i,j}$ são visíveis ao escalonador e podem ser utilizados para decidir a prioridade dos casos.

5.3.3 *Allowance Factor* (A)

O *allowance factor* é um valor que determina a quantidade de tempo que um caso recebe para executar sem estar atrasado. Desta forma, a variável A regula a janela de tempo de execução do caso. Por exemplo, se o valor de $A = 2$ e o tempo de processamento esperado for $\overline{p}'_j = 100$, então o caso terá um prazo igual a $200 + r_j$, ou seja, o caso terá duas vezes seu tempo mínimo esperado de execução para que possa ser concluído sem que seja considerado um caso atrasado. Desta forma, quando A for baixo, diz-se que o cenário é mais carregado (possui prazos mais curtos), e quando for alto, diz-se que ele é menos carregado (prazos mais longos).

5.4 Gerenciamento das Filas

O conceito de carga de trabalho está diretamente relacionado com o número de recursos processantes e com o tamanho das filas existentes [4]. A importância de tamanho das filas influenciam diretamente as políticas de escalonamento. Esse fato é observado quando se está trabalho com sistemas superdimensionados, ou seja, muitos recursos disponíveis, pouco trabalho a ser realizado e a inexistência de filas. Nesta situação, o potencial de otimização do escalonador se desfaz, pois o sentido de prioridade e a redução de tarefas atrasadas não são mais relevantes, dado que todos os casos serão executados assim que chegarem ao sistema.

Para aumentar o escopo de otimização do escalonador, optou-se por vincular a carga do sistema com o tamanho das filas. A carga do sistema ou *workload* é representada pelos atributos U e A; assim, para se obter um tamanho médio para as filas do sistema, esses valores devem ser atribuídos por meio de medições empíricas em cada um dos cenários avaliados.

O intuito dessa vinculação é garantir a existência de um número mínimo de casos dentro de uma fila, aumentando o escopo de escolhas do escalonador quando o mesmo define as prioridades de cada caso. Desse modo, considerando um número de casos igual a 200, foi estabelecida empiricamente que uma média de tamanho de fila igual a 5 como sendo um valor mediano. Assim, valores médios de filas maiores que 5 seria sistemas fortemente carregados, e valores menores que 5 representam sistemas levemente carregados.

Para se manter um constância maior no tamanho das filas, a mudança dinâmica do valor de U poderia ter sido implementada. Porém, caso esta abordagem tivesse sido escolhida, o comportamento de chegada dos casos de um sistema real, sugerido por Laguna e Marklunk [25], não seria preservado.

5.5 Métricas de Avaliação

Pode-se pensar que um sistema de *workflow* é eficiente quando consegue processar um grande número de casos em um curto período. Porém, a eficiência de um sistema está diretamente ligada à função objetivo que pretendemos otimizar.

Neste sentido, dois princípios foram utilizados para definir as métricas de avaliação. O primeiro, relacionado com os prazos dos casos e, o segundo, relacionado com o tempo de processamento. Para quantificar o atraso, foi utilizada a porcentagem média de casos atrasados e a porcentagem média de atraso. Já a métrica utilizada para avaliar o tempo de processamento foi o tempo médio de processamento.

O tempo médio de processamento, como o próprio nome indica, mede o tempo médio que um caso leva para ser executado no sistema. Minimizar esta métrica significa fazer com que os casos sejam executados mais rápidos, reduzindo o tempo médio de processamento. O tempo médio de processamento de um caso no sistema é uma métrica relativamente simples, representado pela Equação 5.7.

$$M_{pt} = \frac{1}{n} \sum_{j=1}^{n} (C_j - r_j) \tag{5.7}$$

A porcentagem média de casos atrasados foi efetuada por meio de uma razão entre o número de casos atrasados e o número total de casos (Equação 5.8).

$$Q = \frac{\sum_{j=1}^{n} I_j}{n} \tag{5.8}$$

Quando o objetivo é fazer com que os casos sejam executados o mais breve possível, sem preocupações extras, o tempo médio de processamento é uma boa métrica. Porém, quando a preocupação está no número de casos atrasados, existindo uma penalidade fixa para cada caso atrasado, a porcentagem média de casos atrasados é mais adequada.

Existem também situações onde a quantidade de cada atraso é relevante. Por exemplo, em um voo que estava planejado para sair de sua origem às 8:00 e, por algum imprevisto, decola às 15:00. Essa situação de atraso, em um ambiente aeroviário deve ser considerada mais relevante do que a de um voo com atraso de 10 minutos. Para esses casos, o atraso médio por casos atrasado é uma métrica mais adequada. Esta última é definida pela Equação 5.9.

$$R = \frac{\frac{1}{n_t}\sum_{j=1}^{n} T_j}{M_{pt}} \tag{5.9}$$

onde:

$n_t = \sum_{j=1}^{n} I_j$, e corresponde ao número de casos atrasados.

5.6 Simulação

Segundo Banks *et al.* [2], simulação é uma técnica utilizada para imitar o comportamento de um sistema ou processo real. O uso da simulação é útil no estudo e na modelagem de processos de negócio, onde o desenvolvimento de modelo matemático é muito difícil ou impossível de ser realizado. Além disso, a análise quantitativa em ambientes dinâmicos, exige um grande conjunto de atributos, os quais podem ser capturados mais facilmente em um ambiente de simulação controlado [25].

A simulação também é capaz de cobrir certas ineficiências que geralmente passam despercebidas até que o sistema venha entrar em operação [25]. A variabilidade de cenários que podem ser avaliados em um curto espaço de tempo é um ponto valioso quando uma decisão sobre o processo deve ser tomada, diminuindo custos de reengenharia no futuro.

Existem diferentes tipos de modelos para simulação. A simulação utilizada neste trabalho representa a evolução dinâmica do sistema por meio de eventos discretos, classificada como *discrete-event simulation*, ou simulação com eventos discretos. Ou seja, o sistema modelado representa a mudança de estados por meio de instantes discretos, onde cada mudança de estado ocorre em decorrência de um certo evento.

Com a finalidade de verificar o desempenho das diferentes políticas de escalonamento em um *workflow*, um conjunto de simuladores foi implementado, um para cada cenário. A construção dos simuladores foi feita com o auxílio do ambiente computacional $MatLab^{®}$. As implementações representam os eventos como sendo unidades inteiras de tempo, retirando a dependência entre as métricas de avaliação e a velocidade do computador que executa o simulador.

5.7 Parâmetros Utilizados

A execução do conjunto de simulações, para cada um dos três cenários, obedece a um conjunto de parâmetros. Os valores desses atributos são definidos nesta sessão.

Atividades: O comportamento de cada atividade é representado por um conjunto de distribuições as quais são atribuídas de forma aleatória a cada uma das atividades. Neste quesito, duas situações foram avaliadas:

- atividades homogêneas: refere-se a existência de atividades com comportamento semelhantes, ou seja, distribuições parecidas; nesta situação somente as distribuições $Dist_{\mu,\sigma} = \{(30, 10); (40, 15)\}$ são utilizadas;
- atividade "pesada": nesta situação, além das distribuições utilizadas na situação homogênea, uma nova distribuição com uma média maior é atribuída a uma das atividades (média $\mu = 150$ e desvio padrão $\sigma = 20$), ou seja, uma das atividades terá um comportamento mais lento; o objetivo dessa atribuição é representar gargalos no sistema paralelo;

Número de Casos: Em cada um dos testes um $n = 200$, ou seja, um conjunto 200 casos diferentes é processado;

Allowance Factor: Este parâmetro foi fixado $A = 2$;

Tamanho Médio das Filas: Parâmetro fixado em 5 unidades, escolhido empiricamente para permitir uma maior otimização do escalonador. com base na carga do sistema;

Carga do Sistema: A atribuição desse parâmetro é dependente do tamanho médio das filas, sendo atribuído de acordo com três classes: leve, um valor de U é atribuído dinamicamente para que a média de filas esteje entre 2 e 4; médio, um valor atribuído para que a média de filas seja igual a 5; e pesado, um valor é atribuído para que a média de filas seja maior que 5 e menor que 10;

Erro Máximo do Previsor os valores aproximados de tempo de execução $p'_{i,j}$ são atribuídos respeitando uma margem de erro igual a $\int_p = 10\%,\ 30\%\ e\ 50\%$;

Número de Execuções para confirmar o comportamento de um dos cenários, 10 execuções são realizadas; em cada nova execução um novo conjunto de casos é utilizado;

Os resultados obtidos pela simulação trazem, em termos quantitativos, um comparativo entre a utilização de técnicas de escalonamento projetadas para estruturas paralelas e a simples aplicação de técnicas mais conhecidas. A comparação desses resultados foi efetuada segundo dois critérios:

- Os resultados numéricos mostrados em cada métrica.
- Os resultados da Análise de Variância para cada conjunto de resultados.

5.8 Análise de Variância e Tukey

A análise de variância (ANOVA) é um teste estatístico amplamente difundido entre os analistas, e visa fundamentalmente verificar se existe uma diferença significativa entre as médias e se os fatores exercem influência em alguma variável dependente [21].

Os fatores propostos podem ser de origem qualitativa ou quantitativa, mas a variável dependente necessariamente deverá ser contínua.

Existem dois métodos para calcular a variância: dentro de grupos (MQG, ou média quadrada dos grupos) e a variância das médias (MQR, ou média quadrada dos resíduos). Em um teste ANOVA, calcula-se esses dois componentes de variância. Se a variância calculada usando a média (MQR) for maior do que a calculada usando (MQG) os dados pertencentes a cada grupo, isso pode indicar que existe uma diferença significativa entre os grupos [21].

Porém, o método ANOVA apenas identifica se existe ou não uma diferença significante. Para verificar o qual grupo é significantemente diferente o teste proposto por Tukey pode ser aplicado.

O Teste proposto por Tukey, também conhecido como teste de Tukey HSD (*Honestly Significant Difference*) é um teste exato em que permite estabelecer a diferença mínima significante, ou seja, a menor diferença de médias de amostras que deve ser tomada como estatisticamente significante, em determinado nível [21].

Neste trabalho, a técnica de geração de resultados consistiu em executar várias simulações para cada combinação de variáveis dos cenários, e depois analisar tais resultados numericamente (valores médios) e estatisticamente. A diferença estatística entre os grupos foi calculada por meio de uma análise de variância e posteriormente pelo teste de Tukey HSD (*Honestly Significant Difference*), ambos implementados pela função *multcompare* do $MatLab^{®}$.

Capítulo 6

Resultados e Discussão

Neste capítulo são apresentados e discutidos os resultados obtidos pelas simulações em cada um dos cenários abordados. As simulações foram executadas de acordo com a descrição efetuada na seção 5.7 e avaliadas segundo as métricas descritas na seção 5.5. Além das médias fornecidas pelas métricas, a comparação dos resultados entre as políticas de escalonamento foi efetivada por meio de uma análise de variância (ANOVA) seguido por um teste de Tukey. Este último foi realizado para determinar se os resultados obtidos podem ser considerados significantemente distintos.

A heurística *Parallel* FIFO-*DueDate* (Par-FIFO-DD), que tem por objetivo reduzir o número de casos atrasados, foi comparada com as políticas EDD e FIFO, pois a EDD possui a mesma meta. Já as heurísticas *Synchronized*-SPT (Sync-SPT) e *Process Time Ratio* (PT-Ratio) possuem como meta reduzir o tempo médio de processamento e foram comparadas com SPT e FIFO. A regra FIFO está presentes em todas as simulações, pois ela foi a base de referência para os resultados das outras técnicas, já que, como mencionado anteriormente, é a política mais comum nos sistemas de *worklfow* atuais. Um diagrama, demonstrando as comparações feitas, é apresentado na Figura 6.1.

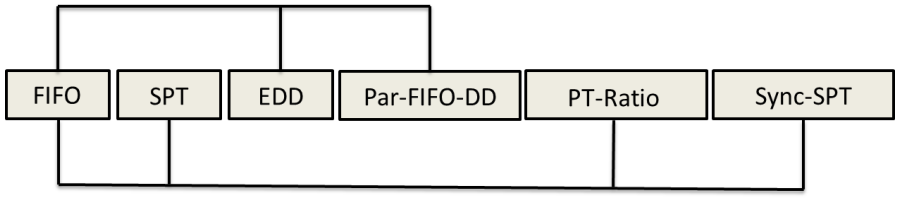


Figura 6.1: Esquema das comparações entre as diferentes políticas de escalonamento.

As técnicas SPT, PT-Ratio, Sync-SPT e a Par-FIFO-DD, realizam o escalonamento com base nos tempos previstos de execução, sendo esses submetidos a um acréscimo de erro (seção 5.3.2). Um sistema de siglas foi adotado para representar tais erros. Por

exemplo, sigla SPT(10) representa os resultados obtidos pela técnica de escalonamento SPT com um erro máximo agregado aos tempos de execução de 10%.

Em cada simulação, um conjunto de 200 casos passam pelo sistema. Este conjunto é testado para cada uma das técnicas, e os resultados são colhidos e armazenados. Cada teste consiste em realizar 10 execuções, onde em cada execução um novo conjunto de casos é instanciado.

Os resultados de todos os testes serão demonstrados por meio de Tabelas, porém uma análise mais detalhada dos resultados com erro máximo igual a 30% é realizada. A escolha pelos resultados com erro máximo 30% se deve a dois fatores: o primeiro é que a redução da margem de erro para 10%, ou o aumento para 50% resulta, respectivamente, em uma pequena melhora ou degradação dos resultados obtidos, mantendo o mesmo comportamento quando o erro é restrito a 30%; o segundo é que estudos mostram que 30% de erro é uma situação mais provável de ocorrer em ambientes reais [41].

6.1 Cenário - 1

A execução eficiente de certo trabalho muitas vezes exige a divisão em partes para que duas ou mais pessoas possam conclui-lo mais rapidamente. Em modelos de *workflow*, muitas dessas situações são representadas por pequenos ANDs, onde um número pequeno de atividades é designado a cada uma das partes. O Cenário - 1 (Figura 6.2) representa essas situações, e os resultados obtidos são apresentados nas subseções seguintes.

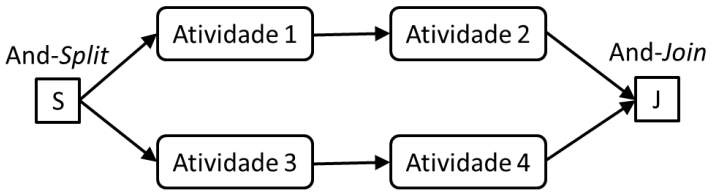


Figura 6.2: Representação gráfica do Cenário-1.

6.1.1 Tempo Médio de Processamento

A Tabela 6.1 apresenta os tempos médios de processamento para o Cenário - 1 em um ambiente de atividades homogêneas. Os melhores resultados foram atribuídos à política *Process Time Ratio*. O resultado mais significante foi em um sistema com carga elevada. Esta situação ocorre devido à existência de filas com tamanhos suficientes para que a ordenação efetuada pela política possa operar eficientemente, resultando na redução do tempo médio de processamento. Conforme a carga aumenta o ganho aumenta. Este ganho é mais facilmente visualizado por meio da Figura 6.3.

Porém, quando os testes ANOVA e Tukey são aplicados, percebe-se que as médias entre FIFO e PT-Ratio satisfazem a hipótese nula, ou seja, as médias não são estatisticamente diferentes, com um grau de 95% de confiança. As diferenças entre as médias e seus intervalos de confiança são representadas pela Figura 6.5. Está Figura mostra que há uma diferença significativa apenas entre as médias da técnica SPT e PT-Ratio.

A técnica Sync-SPT demonstra bons resultados quando comparada a SPT, porém quando comparada com a política FIFO os resultados são menos representativos. A heurística Sync-SPT pode ser interpretada como uma adequação da SPT para estruturas paralelas (AND), utilizando a correlação das partes de um mesmo caso, e permitindo um melhor sincronismo entre os casos em execução. A melhora dos resultados da Sync-SPT em relação a SPT é relevante. Esses demonstram que o desenvolvimento de políticas específicas de escalonamento para estruturas AND pode contribuir para uma melhora na eficiência de sistemas de *workflow*.

Os resultados apresentados na Tabela 6.2 são referentes a um cenário onde uma das atividades possui uma distribuição mais "pesada" (seção 5.7). A atribuição desta distribuição permite a inserção de um gargalo em um dos canais do AND, dificultando o sincronismo das partes de um mesmo caso. Apesar do comportamento das políticas ser semelhante, pouca diferença é observada em relação às diferentes cargas do sistema, fato causado pela atividade "pesada", que adiciona um tempo alto de processamento a cada uma das atividades. Um resumo desses resultados pode ser visto na Figura 6.4.

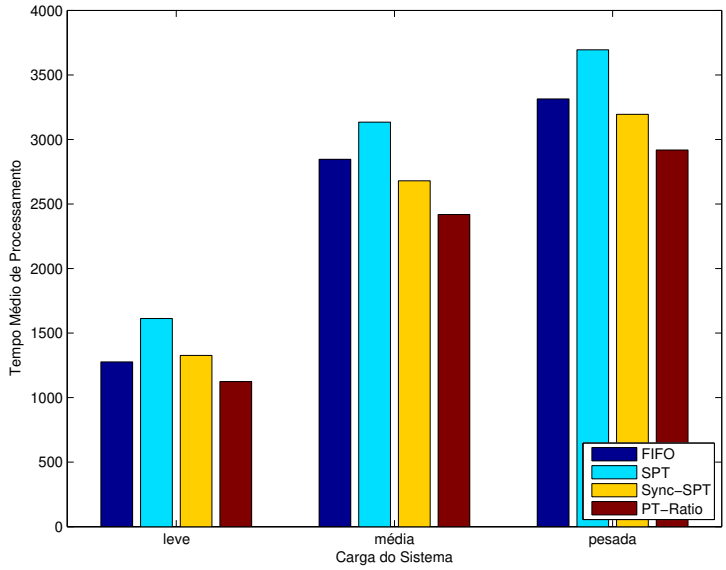


Figura 6.3: Tempo médio de processamento: cenário - 1; atividades homogêneas; erro máximo de 30%.

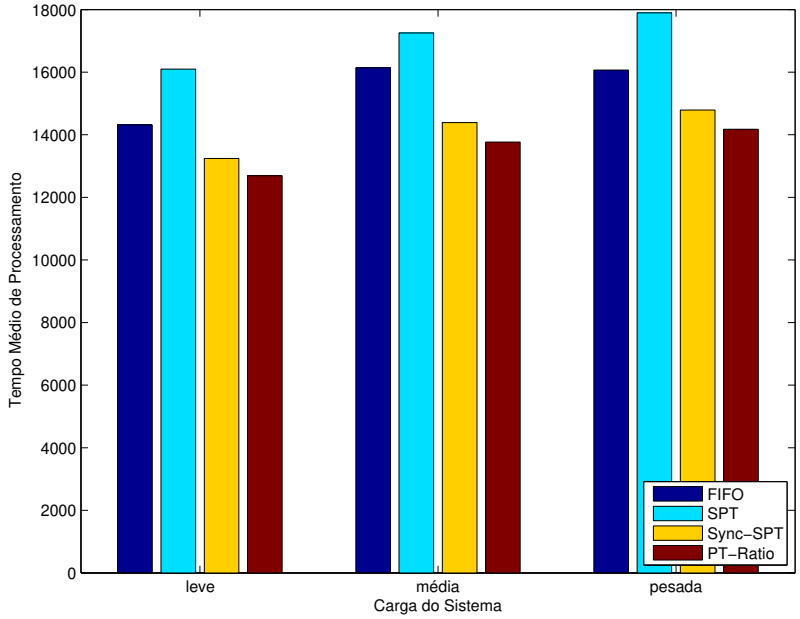


Figura 6.4: Tempo médio de processamento: cenário - 1; uma atividade com uma distribuição "pesada"; erro máximo é igual a 30%.

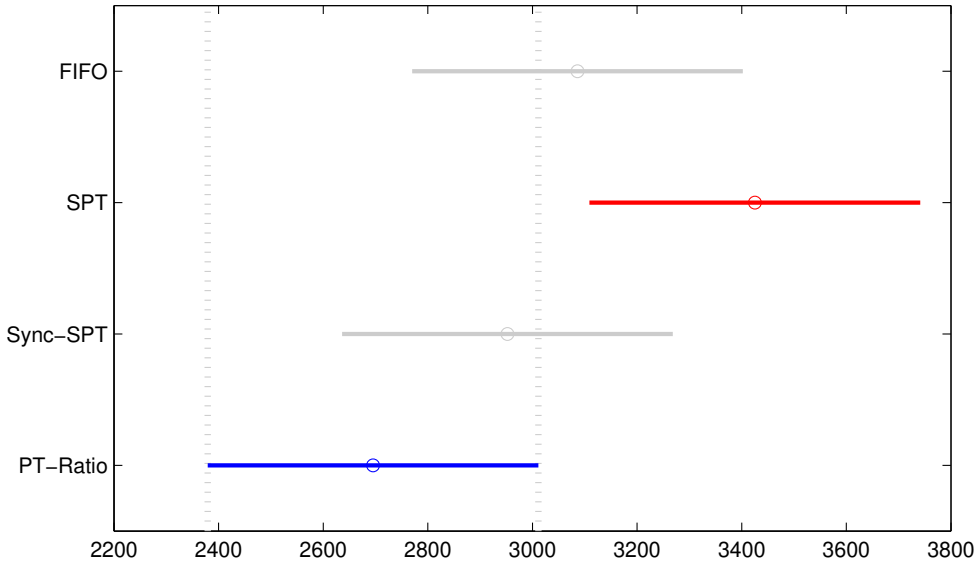


Figura 6.5: Avaliação por meio da ANOVA com Tukey para o tempo médio de processamento: cenário - 1; atividades homogêneas; erro máximo de 30%; sistema com uma carga "pesada".

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	1276,225	2845,999	3314,385
SPT(10)	1583,84	3077,786	3585,89
SPT(30)	1612,637	3133,746	3694,5533
SPT(50)	1843,013	3581,424	4129,2067
PT-Ratio(10)	1054,146	2245,922	2765,16
PT-Ratio(30)	1124,423	2418,685	2918,78
PT-Ratio(50)	1236,865	2660,554	3162,0117
Sync-SPT(10)	1238,291	2545,432	2995,4109
Sync-SPT(30)	1326,74	2679,403	3195,105
Sync-SPT(50)	1459,414	2947,343	3514,6155

Tabela 6.1: Resultado: tempo médio de processamento; cenário - 1; atividades homogêneas.

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	14313,83	16136,3	16066,04
SPT(10)	15809,97	16945,75	17369,32
SPT(30)	16097,43	17253,86	17895,67
SPT(50)	18397,06	19718,69	20001,04
PT-Ratio(10)	11895,53	12782,54	13428,53
PT-Ratio(30)	12688,57	13765,82	14174,56
PT-Ratio(50)	13957,42	15142,4	15355,77
Sync-SPT(10)	12360,44	13668,08	13866,56
Sync-SPT(30)	13243,33	14387,45	14791
Sync-SPT(50)	14567,67	15826,19	16270,1

Tabela 6.2: Resultados: tempo médio de processamento; cenário-1; uma atividade com distribuição “pesada”.

6.1.2 Atraso de Casos

Os resultados obtidos para um conjunto de atividades homogêneas é apresentado nas Tabelas: 6.3, 6.4, 6.5 e 6.6. Um resumo desses resultados é apresentado na Figura 6.6. Para o ambiente que contém uma atividade "pesada", as seguintes Tabelas listam os resultados obtidos: Tabela 6.7, Tabela 6.8 e Tabela 6.9. Um resumo desses últimos resultados é apresentado na Figura 6.7.

Analisando os resultados do conjunto de atividades homogêneas, nota-se um alto número de casos atrasados, mesmo em situações de baixa carga no sistema. Isso se deve a dois fatores: o primeiro é o fato do cálculo do *due date* ser atribuído com base no tempo mínimo de execução necessário para um caso ser concluído; assim, como o Cenário - 1 possui uma profundidade de 2 (apenas 2 atividades em série), este tempo é reduzido; o segundo está relacionado ao *allowance factor* estar fixado em 2.

Com relação ao número de atrasos, os resultados foram mais expressivos que os aferidos no número de atrasos. A heurística Par-FIFO-DD obteve os melhores resultados, sendo mais expressivos em um ambiente homogêneo de atividades e com cargas leves (Figura 6.6). Porém, o ganho na redução do número de atrasos se reflete diretamente em um aumento na média de atraso, ocasionado pela prioridade que a heurística Par-FIFO-DD dá à situações onde os casos estão prestes a atrasar. Apesar do aumento da média de atraso, a redução do número de casos atrasados é relevante, fato comprovado no testes ANOVA e Tukey(Figura 6.8).

Já em situações onde uma das atividades possui um comportamento mais lento (distribuição mais "pesada" é atribuída), um aumento do número de casos atrasados é percebido (Figura 6.7). Apesar dessa redução, a avaliação dos resultados pela ANOVA mostra que a política Par-FIFO-DD é mais eficiente. Nesta situação, o atraso médio por caso atrasado da Par-FIFO-DD é menor que a EDD.

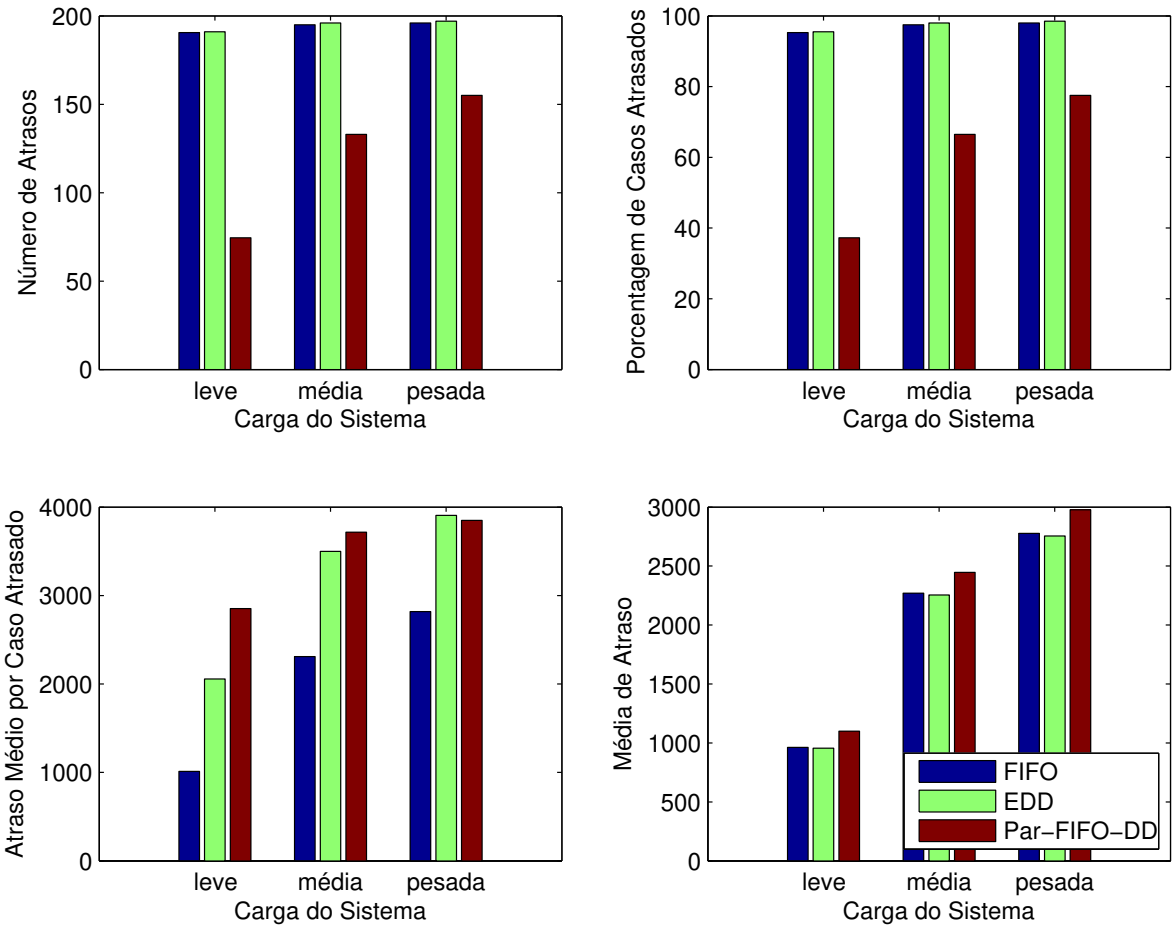


Figura 6.6: Representação gráfica do número de atrasos médios, porcentagem de casos atrasados, atraso médio por caso atrasado e média de atraso. Dados referentes a simulação: cenário - 1; atividades homogêneas; erro máximo de 30%.

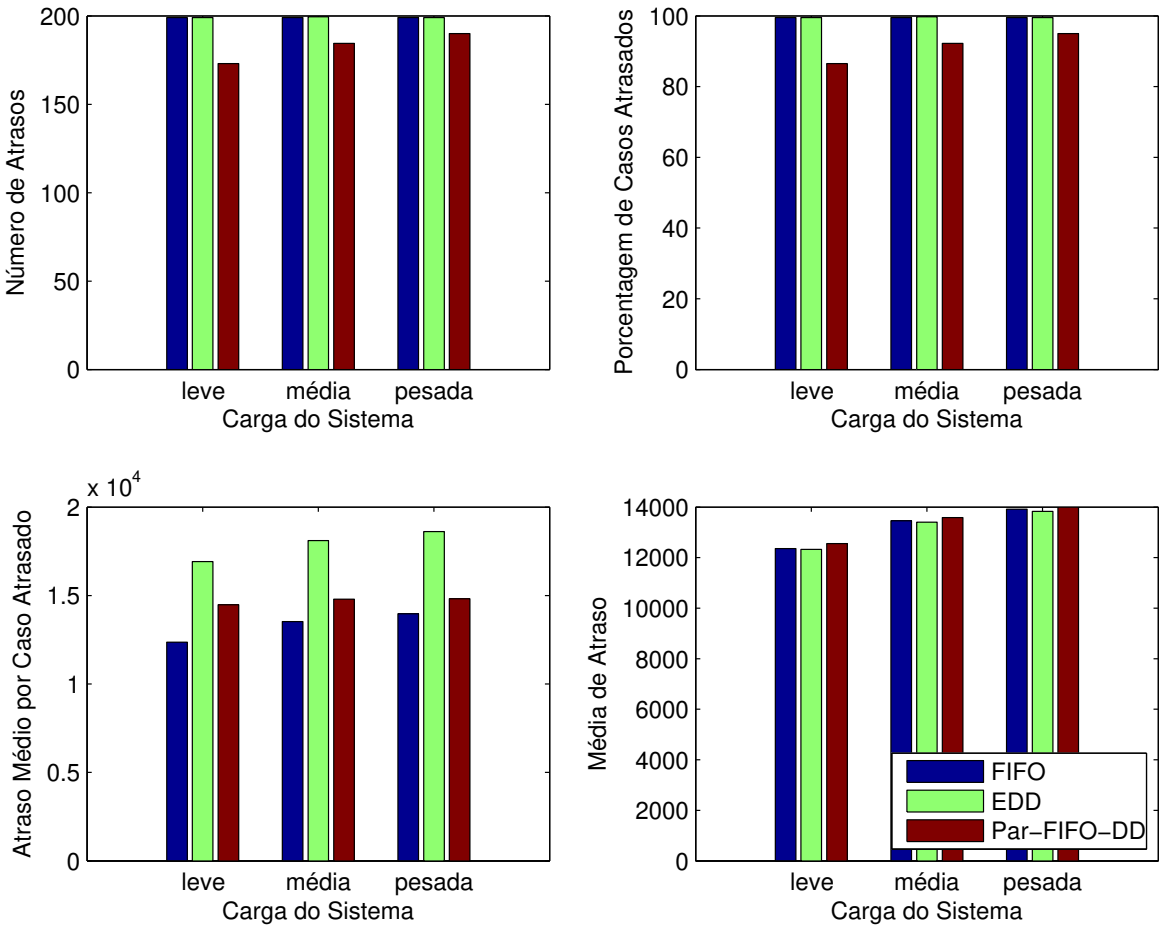


Figura 6.7: Representação gráfica do número de atrasos médios, porcentagem de casos atrasados, atraso médio por caso atrasado e média de atraso. Dados referentes a simulação: cenário - 1; uma atividade com distribuição "pesada"; erro máximo de 30%.

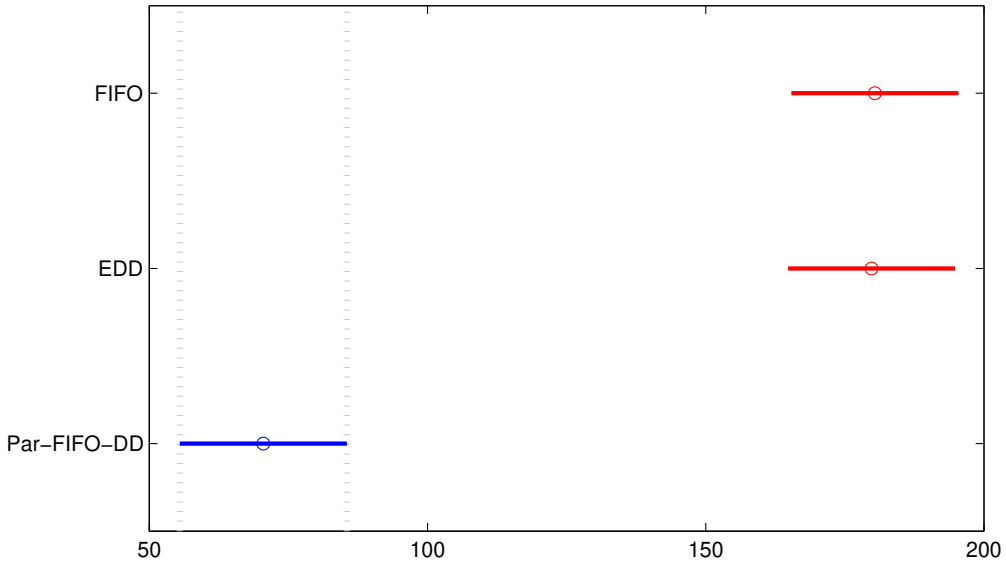


Figura 6.8: Avaliação por meio da ANOVA com Tukey para o número de atrasos: cenário - 1; atividades homogêneas; erro máximo de 30%; sistema com uma carga "pesada".

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	190,5	195	196
EDD	191	196	197
Par-FIFO-DD(10)	73,5	132	153,5
Par-FIFO-DD(30)	74,5	133	155
Par-FIFO-DD(50)	76	132,5	152,5

Tabela 6.3: Resultados: número médio de atrasos; cenário - 1, atividades homogêneas.

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	962,595	2270,028	2777,15
EDD	955,5175	2254,053	2754,578
Par-FIFO-DD(10)	1059,548	2402,755	2998,255
Par-FIFO-DD(30)	1099,14	2447,008	2977,703
Par-FIFO-DD(50)	1045,013	1045,013	2823,438

Tabela 6.4: Resultados: atraso médio; cenário - 1; atividades homogêneas.

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	1011,254	2310,517	2819,442
EDD	1001,43	2299,946	2796,373
Par-FIFO-DD(10)	2895,445	3619,942	3901,808
Par-FIFO-DD(30)	2852,484	3716,731	3849,906
Par-FIFO-DD(50)	2661,127	3569,431	3631,913

Tabela 6.5: Resultados: atraso médio por casos atrasado; cenário - 1, atividades homogêneas.

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	95,25	97,5	98
EDD	95,5	98	98,5
Par-FIFO-DD(10)	36,75	66	76,75
Par-FIFO-DD(30)	37,25	66,5	77,5
Par-FIFO-DD(50)	38	66	76,25

Tabela 6.6: Resultados: porcentagem de atrasos; cenário - 1; atividades homogêneas.

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	162,5	183,5	188
EDD	163,5	184	186,5
Par-FIFO-DD(10)	49	119	138
Par-FIFO-DD(30)	49,5	121	144
Par-FIFO-DD(50)	53	132	150

Tabela 6.7: Resultados: número médio de atrasos; cenário - 1, uma atividade com distribuição "pesada".

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	386,9175	386,9175	1849,233
EDD	377,7375	1449,17	1816,093
Par-FIFO-DD(10)	794,7594	1756,224	2009,649
Par-FIFO-DD(30)	645,5225	1655,113	1978,575
Par-FIFO-DD(50)	629,925	1352,113	1814,477

Tabela 6.8: Resultados: atraso médio ; cenário - 1, uma atividade com distribuição "pesada".

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	498,3204	1620,115	1988,598
EDD	495,0803	1602,246	1958,059
Par-FIFO-DD(10)	2832,542	3002,306	2999,036
Par-FIFO-DD(30)	2775,938	2811,154	2874,344
Par-FIFO-DD(50)	2456,942	2221,931	2749,804

Tabela 6.9: Resultados: atraso médio por caso atrasado; cenário - 1, uma atividade com distribuição "pesada".

Técnica	Carga do Sistema		
	leve	*médio*	*pesado*
FIFO	81,25	91,75	94
EDD	81,75	92	93,25
Par-FIFO-DD(10)	24,5	59,5	69
Par-FIFO-DD(30)	24,75	60,5	72
Par-FIFO-DD(50)	26,5	66	75

Tabela 6.10: Resultados:porcentagem de atraso; cenário - 1, uma atividade com distribuição "pesada".

6.2 Cenário - 2

Este cenário tem por objetivo representar situações extremas onde um dos canais do AND é muito maior que o outro, dificultando a tarefa de manter o o sincronismo dos casos. Apesar de extremo, este cenário é comum em ambientes de *workflow* (Figura 6.9).

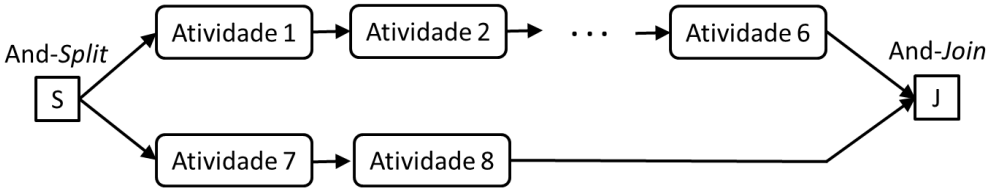


Figura 6.9: Representação gráfica do Cenário-2.

6.2.1 Tempo Médio de Processamento

Para cargas leves e medianas, as políticas de escalonamento tiveram um comportamento semelhante, porém com uma leve vantagem para a SPT. Os melhores resultados para a Sync-SPT e PT-Ratio foram obtidos em cargas mais pesadas, onde o aumento de filas implica no aumento do escopo de ordenação das técnicas, contribuindo para um leve aumento no desempenho (Figura 6.10 e 6.11).

Analisando os resultados obtidos para o ambiente de atividades homogêneas (Tabela 6.11) por meio dos testes ANOVA e Tukey, constata-se que não há diferença significativa entre os resultados. Esse fato pode ser visto através da Figura 6.12, que mostra as médias e seus respectivos intervalos de confiança.

Para ANDs com grandes diferenças entre a profundidade de seus canais, uma assincronia é gerada. Os resultados obtidos comprovam essa hipótese (Tabela 6.11). Para tentar equilibrar esse ambiente, uma distribuição "pesada" foi inserida no canal de menor profundidade (canal 2: composto pelas atividadea 7 e 8). Porém, os resultados obtidos não foram promissores (Tabela 6.12 e Figura 6.4). A existência de apenas 2 filas no canal menor (filas das atividades 1 e 2) faz com que o escalonador tenha poucas oportunidades para organizar os casos, mantendo a dificuldade de sincronia dos casos.

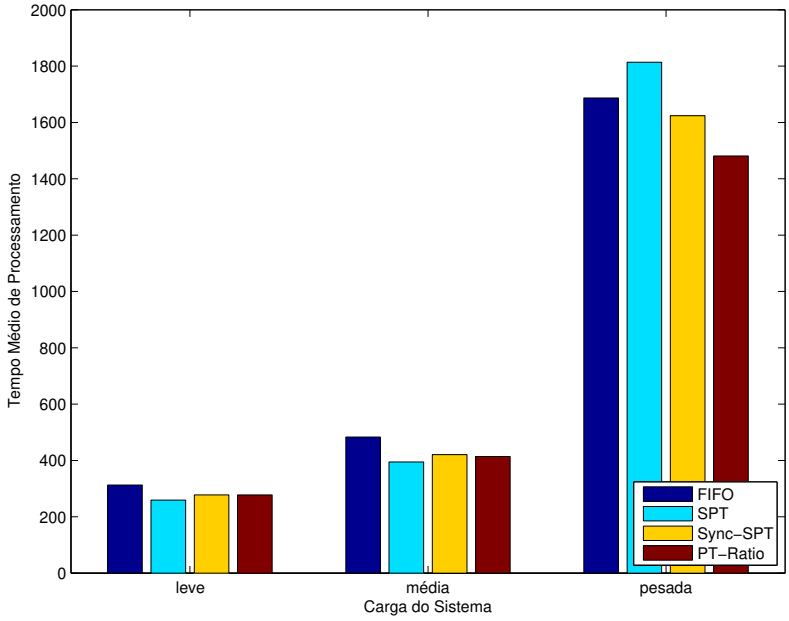


Figura 6.10: Tempo médio de processamento: cenário - 2; atividades homogêneas; erro máximo de 30%.

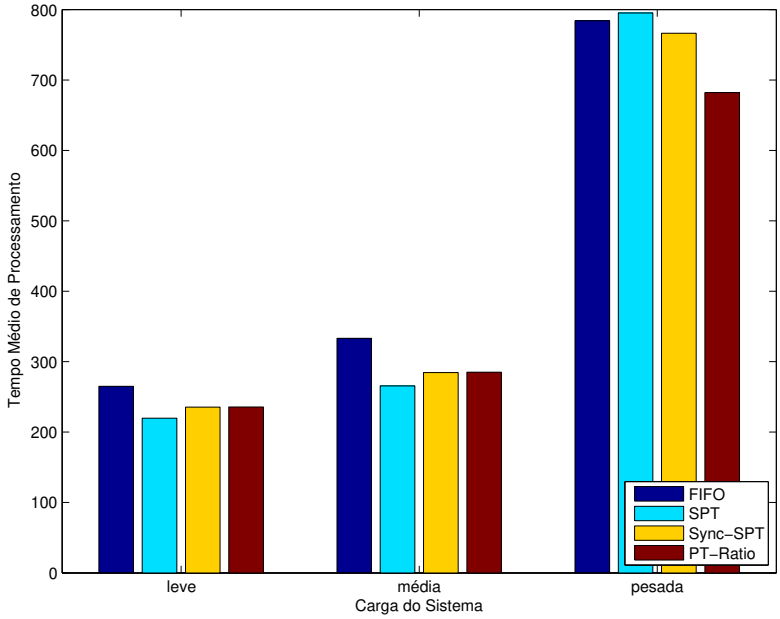


Figura 6.11: Tempo médio de processamento: cenário - 2; uma atividade com uma distribuição "pesada"; erro máximo é igual a 30%.

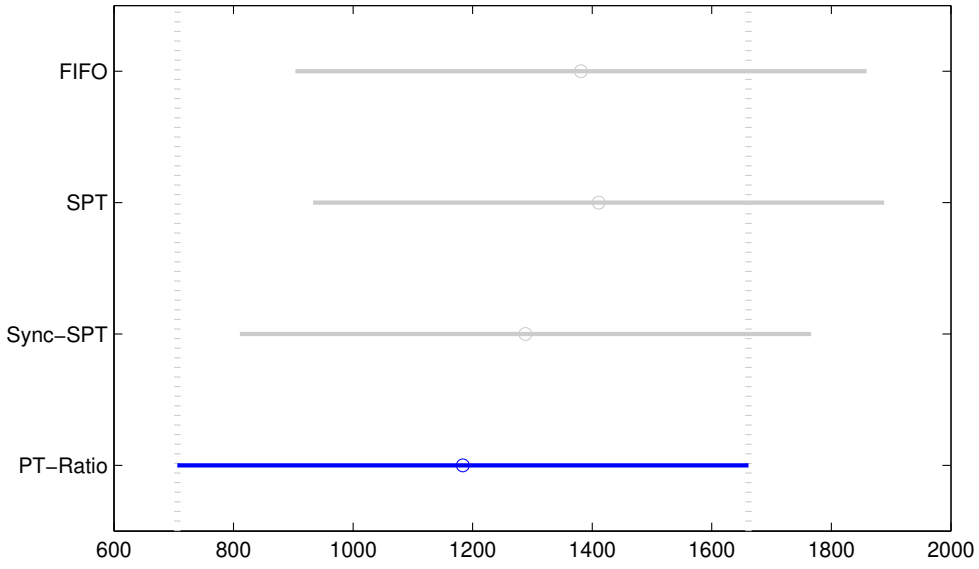


Figura 6.12: Avaliação por meio da ANOVA com Tukey para o tempo médio de processamento: cenário - 2; atividades homogêneas; erro máximo de 30%; sistema com uma carga "pesada".

Técnica	**Carga do Sistema**		
	leve	*média*	*pesada*
FIFO	312,4575	483,0554	1686,839
SPT(10)	254,6454	387,5071	1760,257
SPT(30)	259,2753	394,5527	1813,598
SPT(50)	296,3147	450,9173	2026,962
PT-Ratio(10)	260,318	384,4077	1403,169
PT-Ratio(30)	277,6725	413,9775	1481,123
PT-Ratio(50)	305,4398	455,3753	1604,549
Sync-SPT(10)	259,2753	399,6888	1522,277
Sync-SPT(30)	277,795	420,725	1623,763
Sync-SPT(50)	305,5745	462,7975	1786,139

Tabela 6.11: Resultados: tempo médio de processamento; cenário - 2, atividades homogêneas.

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	265,0022	333,1533	783,9366
SPT(10)	215,8246	260,8306	771,7967
SPT(30)	219,7487	265,573	795,1844
SPT(50)	251,1413	303,512	888,7356
PT-Ratio(10)	220,8352	264,6939	646,2426
PT-Ratio(30)	235,5575	285,055	682,145
PT-Ratio(50)	259,1133	313,5605	738,9904
Sync-SPT(10)	219,7487	270,3154	718,5234
Sync-SPT(30)	235,445	284,5425	766,425
Sync-SPT(50)	258,9895	312,9968	843,0675

Tabela 6.12: Resultados: tempo médio de processamento; cenário - 2, uma atividade com distribuição “pesada”.

6.2.2 Atraso de Casos

Neste cenário, os resultados em um ambiente de atividades homogêneas são apresentados pelas Tabelas: 6.13, 6.15, 6.14 e 6.16. Um resumo dos resultados pode ser visualizado por meio da Figura 6.13.

Em uma situação de carga leve, os resultados são semelhantes, não havendo distinção segundo a ANOVA e Tukey. Já para cargas mais "pesada", uma maior diferenciação é notada (Figura 6.15). Porém, os resultados mais interessantes são observados para cargas medianas. Nesses, a heurística Par-FIFO-DD reduz o número de atrasos e mantém um baixo valor para o atraso médio (Figura 6.13).

Os resultados obtidos para um ambiente com uma distribuição "pesada" são apresentados pelas Tabelas: 6.17, 6.19, 6.18 e 6.20. Um resumo dos resultados pode ser visualizado por meio da Figura 6.14.

Apesar do cenário extremo, a redução do número de atrasos é observada quando a carga do sistema é mais elevada. A inserção de uma distribuição com média mais elevada acarreta um aumento do *due date*, ampliando os prazos e reduzindo o número de atrasos. Os prazos maiores, e com o aumento do tamanho médio das filas, permite que a política Par-FIFO-DD organize melhor os casos, resultando em uma redução no número de atrasos. Em contrapartida, um aumento no atraso médio por caso atrasado é observado.

Mesmo com a grande diferença observada entre a heurística Par-FIFO-DD (Figura 6.14), para o número de amostras igual a 10 (número de execuções), a diferença não é estatísticamente significante (Figura 6.15).

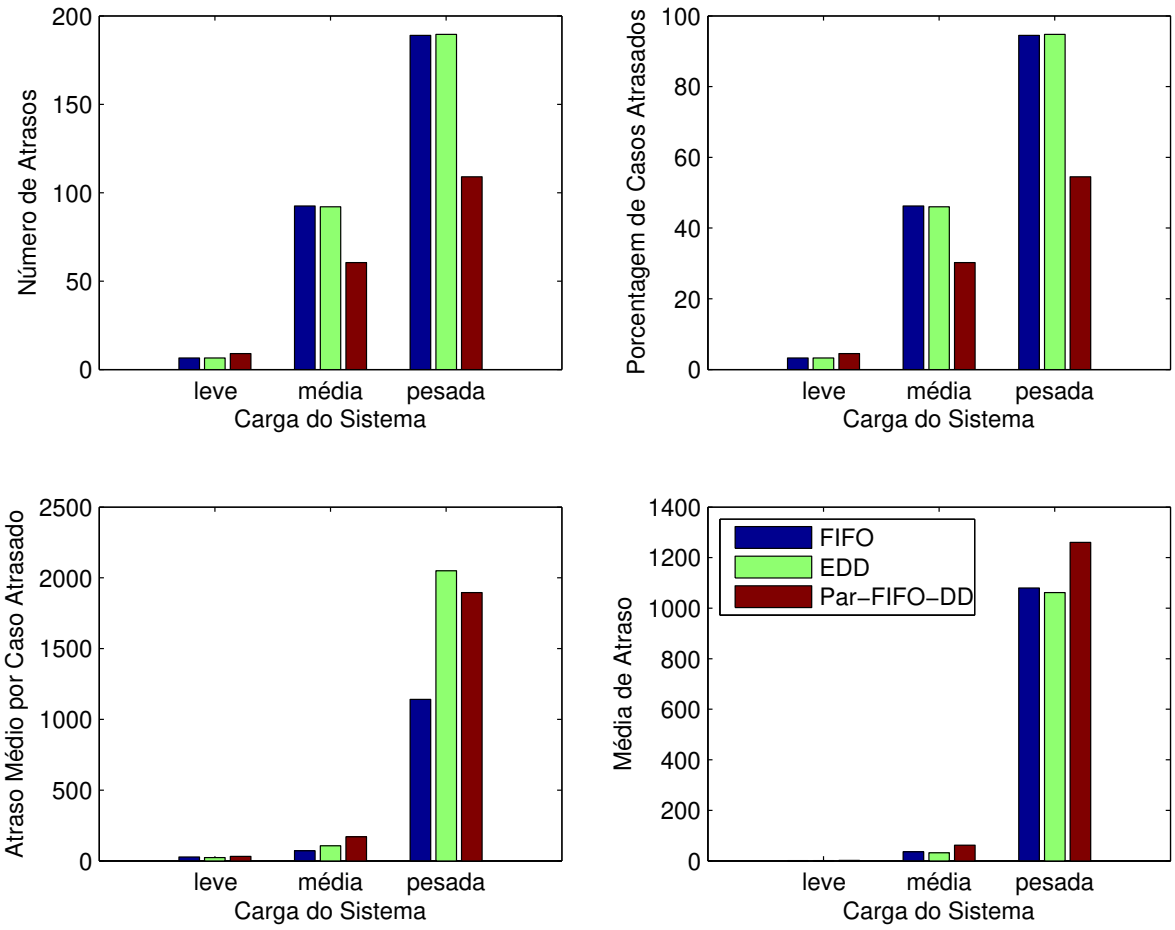


Figura 6.13: Representação gráfica do número de atrasos médios, porcentagem de casos atrasados, atraso médio por caso atrasado e média de atraso. Dados referentes a simulação: cenário - 2; atividades homogêneas; erro máximo de 30%.

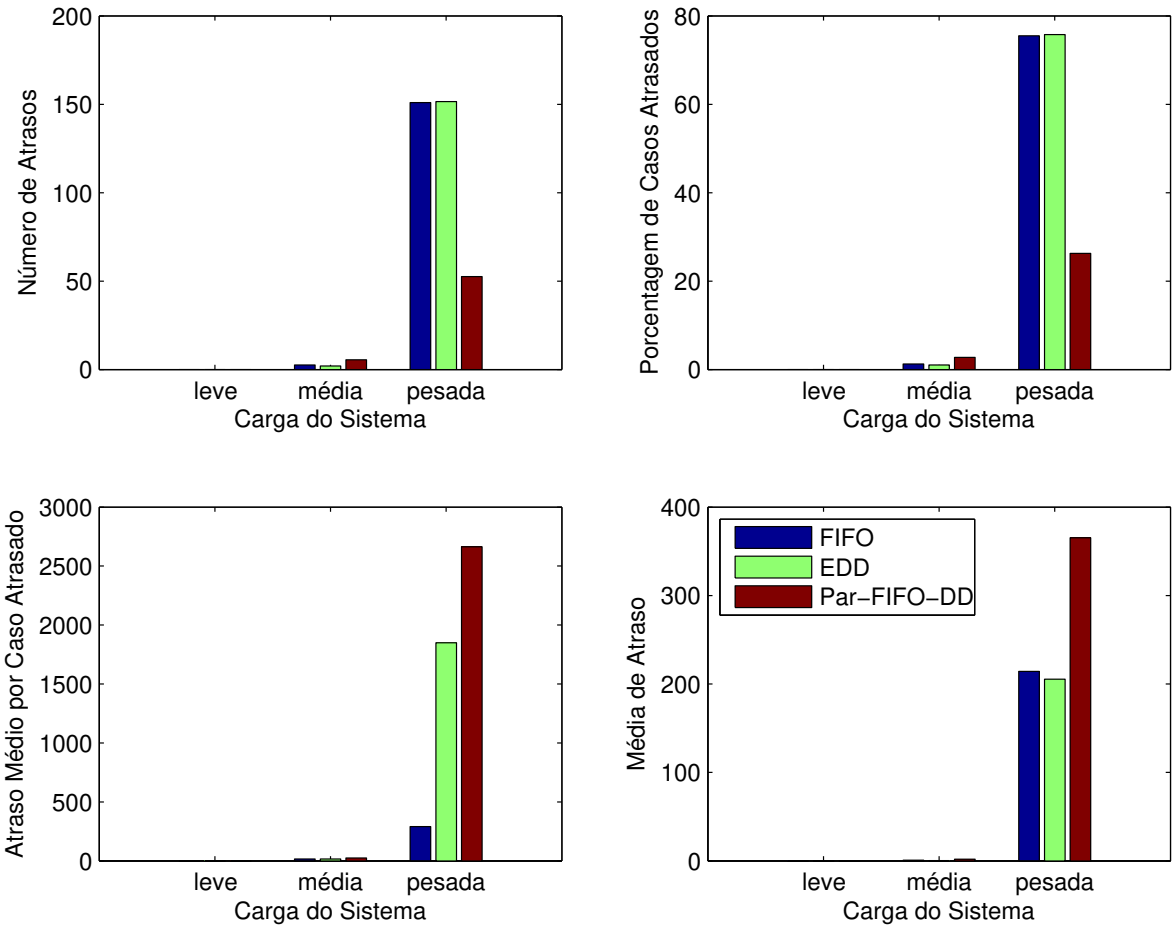


Figura 6.14: Representação gráfica do número de atrasos médios, porcentagem de casos atrasados, atraso médio por caso atrasado e média de atraso. Dados referentes a simulação: cenário - 2; uma atividade com distribuição "pesada"; erro máximo de 30%.

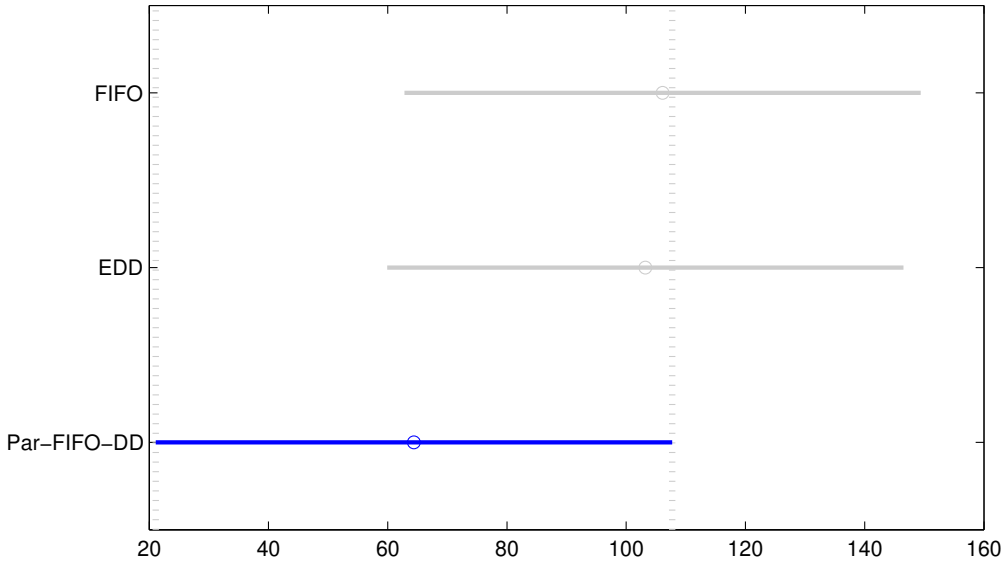


Figura 6.15: Avaliação por meio da ANOVA com Tukey para o número de atrasos: cenário - 2; atividades homogêneas; erro máximo de 30%; sistema com uma carga "pesada".

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	6,5	92,5	189
EDD	6,5	92	189,5
Par-FIFO-DD(10)	8	59,5	98,5
Par-FIFO-DD(30)	9	60,5	109
Par-FIFO-DD(50)	10	64,5	97

Tabela 6.13: Resultados: número médio de atrasos; cenário - 2, atividades homogêneas.

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	1,37	36,3975	1079,973
EDD	1,2725	31,95	1061,82
Par-FIFO-DD(10)	2,0198	69,4257	1589,456
Par-FIFO-DD(30)	1,8225	61,9975	1260
Par-FIFO-DD(50)	2,12	66,5025	1110,793

Tabela 6.14: Resultados: atraso médio; cenário - 2; atividades homogêneas.

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	28,04545	28,04545	1140,499
EDD	25,60795	60,13422	1118,475
Par-FIFO-DD(10)	34,56896	210,5173	2498,546
Par-FIFO-DD(30)	31,46538	169,8827	1896,264
Par-FIFO-DD(50)	37,26538	205,5511	2296,445

Tabela 6.15: Resultados: atraso médio por casos atrasado; cenário - 2, atividades homogêneas.

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	3,25	46,25	94,5
EDD	3,25	46	94,75
Par-FIFO-DD(10)	4	29,75	49,25
Par-FIFO-DD(30)	4,5	30,25	54,5
Par-FIFO-DD(50)	5	32,25	48,5

Tabela 6.16: Resultados: porcentagem de atrasos; cenário - 2; atividades homogêneas.

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	0	2,5	151
EDD	0	2	151,5
Par-FIFO-DD(10)	0	3,5	49,5
Par-FIFO-DD(30)	0	5,5	52,5
Par-FIFO-DD(50)	0	7,2	58,2

Tabela 6.17: Resultados: número médio de atrasos; cenário - 2, uma atividade com distribuição "pesada".

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	0	0,745	214,2475
EDD	0	0,25	205,41
Par-FIFO-DD(10)	0	2,3254	395,5476
Par-FIFO-DD(30)	0	1,9975	365,2525
Par-FIFO-DD(50)	0	1,0026	298,4586

Tabela 6.18: Resultados: atraso médio ; cenário - 2, uma atividade com distribuição "pesada".

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	0	15,40625	290,2291
EDD	0	12,5	274,3353
Par-FIFO-DD(10)	0	26,24589	2846,231
Par-FIFO-DD(30)	0	24,96875	2664,12
Par-FIFO-DD(50)	0	23,4269	2534,131

Tabela 6.19: Resultados: atraso médio por caso atrasado; cenário - 2, uma atividade com distribuição "pesada".

Técnica	Carga do Sistema		
	leve	*médio*	*pesado*
FIFO	0	1,25	75,5
EDD	0	1	75,75
Par-FIFO-DD(10)	0	1,95	24,75
Par-FIFO-DD(30)	0	2,75	26,25
Par-FIFO-DD(50)	0	3,6	29,1

Tabela 6.20: Resultados:porcentagem de atraso; cenário - 2, uma atividade com distribuição "pesada".

6.3 Cenário - 3

O objetivo deste cenário é apresentar uma situação mais propícia às heurísticas de escalonamento propostas (Figura 6.16). Elas utilizam a correlação entre as partes de um mesmo caso para melhorar o sincronismo dentro de um AND. Neste cenário, um conjunto maior de atividades em série é colocado em cada um dos canais do AND, aumentando o número de filas e permitindo ao escalonador corrigir uma atribuição mal sucedida.

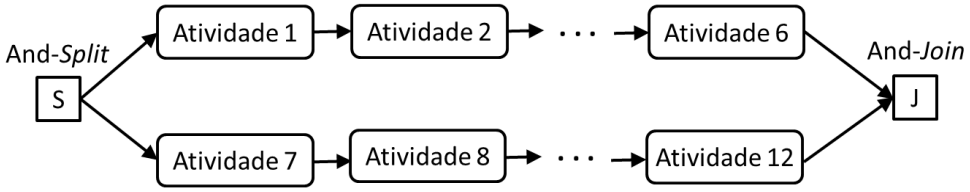


Figura 6.16: Representação gráfica do Cenário-3.

6.3.1 Tempo Médio de Processamento

Neste cenário, a alteração da carga no sistema não mudou o comportamento das diferentes políticas de escalonamento. Os resultados são apresentados nas Tabelas: 6.21 e 6.22. Um resumo dos resultados desses resultados é apresentado nas Figuras: 6.17 e 6.18.

Considerando a situação em que somente atividades com o comportamento semelhante foram inseridas, os melhores resultados foram obtidos pela heurística PT-Ratio em sistemas com carga mediana. A significância desses resultados, avaliada por meio da ANOVA e Tukey, é apresentada na Figura 6.19. Nota-se também que a heurística Sync-SPT apresenta melhores resultados quando comparados à FIFO.

Os resultados obtidos pelas políticas FIFO e SPT foram semelhantes. Apesar da política FIFO manter naturalmente um sincronismo em ambientes homogêneos, ela não obteve bons resultados. Um dos possíveis motivos motivos é que a FIFO não considera os tempos de processamentos necessários para concluir um caso. Em contrapartida, a política SPT trabalha exclusivamente com os tempos de processamento; porém, a falta de correlação entre as partes de um mesmo caso faz com que seu desempenho seja degradado.

Considerando a situação onde existe uma atividade com uma distribuição mais "pesada", um desbalanceamento é criado, degradando ainda mais os resultados obtidos pela política FIFO.

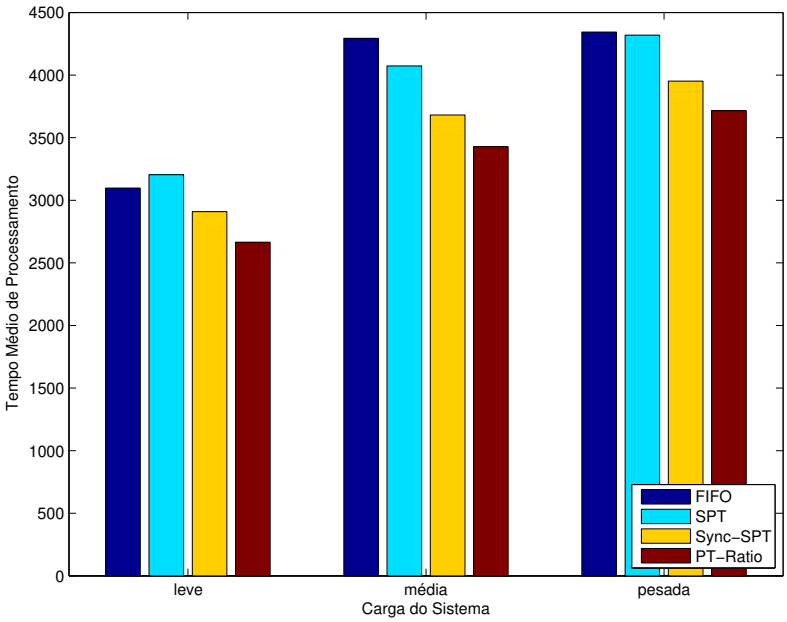


Figura 6.17: Tempo médio de processamento: cenário - 3; atividades homogêneas; erro máximo de 30%.

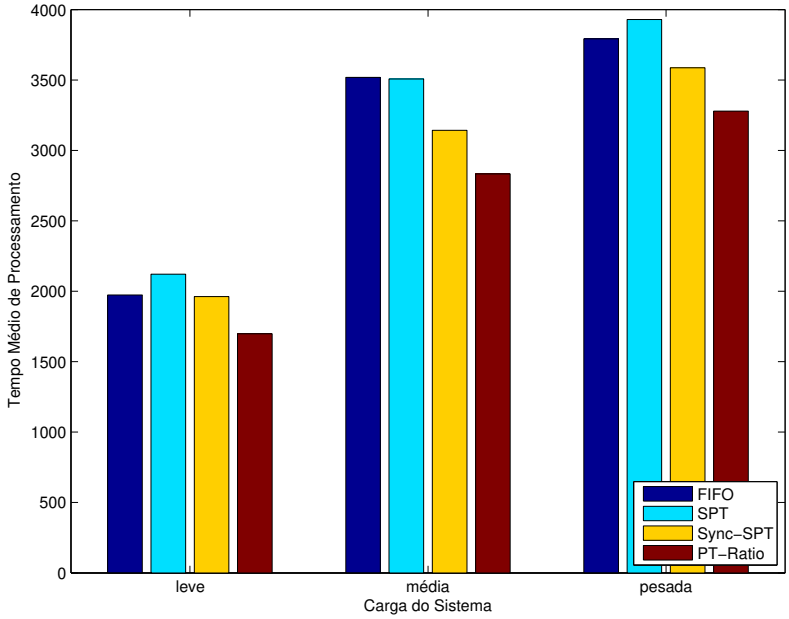


Figura 6.18: Tempo médio de processamento: cenário - 3; uma atividade com uma distribuição "pesada"; erro máximo é igual a 30%.

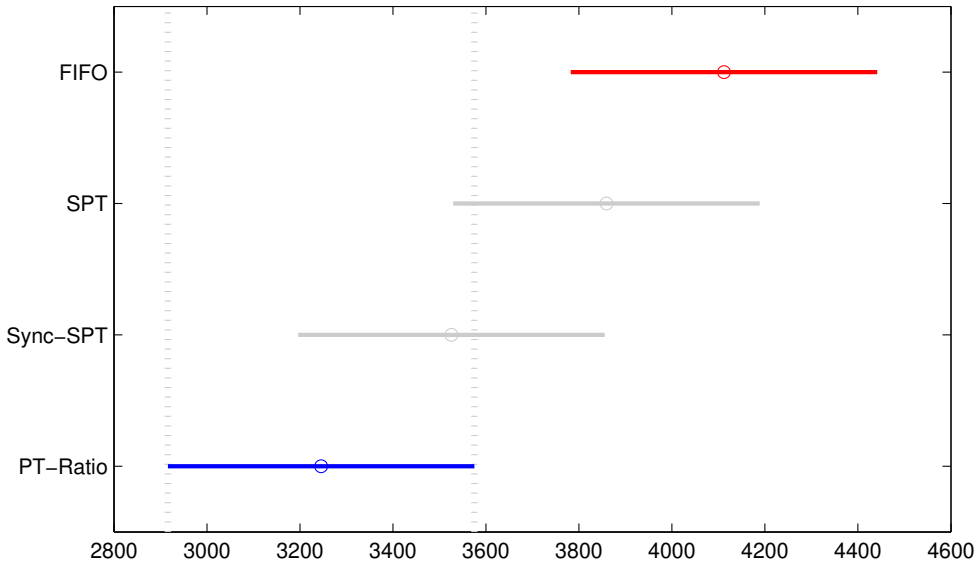


Figura 6.19: Avaliação por meio da ANOVA com Tukey para o tempo médio de processamento: cenário - 3; atividades homogêneas; erro máximo de 30%; sistema com uma carga mediana.

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	3097,637	4294,181	4343,763
SPT(10)	3147,038	4001,09	4186,319
SPT(30)	3204,257	4073,837	4319,218
SPT(50)	3662,008	4655,813	4942,182
PT-Ratio(10)	2498,653	3182,867	3520,234
PT-Ratio(30)	2665,23	3427,703	3715,803
PT-Ratio(50)	2931,753	3770,473	4025,453
Sync-SPT(10)	2715,811	3497,841	3704,845
Sync-SPT(30)	2909,798	3681,938	3951,835
Sync-SPT(50)	3200,777	4050,131	4347,019

Tabela 6.21: Resultados: tempo médio de processamento; cenário - 3, atividades homogêneas.

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	1973,148	3518,784	3792,408
SPT(10)	2083,73	3444,18	3808,643
SPT(30)	2121,616	3506,802	3929,552
SPT(50)	2424,704	4007,773	4496,314
PT-Ratio(10)	1591,945	2630,947	3106,528
PT-Ratio(30)	1698,075	2833,328	3279,113
PT-Ratio(50)	1867,883	3116,66	3552,372
Sync-SPT(10)	1830,992	2985,508	3362,998
Sync-SPT(30)	1961,778	3142,64	3587,198
Sync-SPT(50)	2157,955	3456,904	3945,917

Tabela 6.22: Resultados: tempo médio de processamento; cenário - 3, uma atividade com distribuição “pesada”.

6.3.2 Atraso de Casos

Os resultados obtidos para um ambiente de atividades homogêneas são apresentados pelas Tabelas: 6.23, 6.25, 6.24 e 6.26. Um resumo dos resultados pode ser visualizado por meio da Figura 6.20.

Neste cenário, os resultados foram relevantes quanto à redução do número de atrasos. O melhor resultado foi obtido pela heurística Par-FIFO-DD em situações de carga leve (Figura 6.20). Avaliando o resultado da ANOVA com Tukey, a diferença entre a heurística Par-FIFO-DD e EDD fica mais evidente (Figura 6.22). Além da redução significativa no número de casos atrasados, a média de atraso da Par-FIFO-DD permaneceu próxima a da EDD e da FIFO, como pode ser observado na Figura 6.23.

Esses resultados comprovam a relevância de utilizar as informações referentes às diferentes partes de um mesmo caso dentro de um AND. A correlação das partes contribui para um aumento no sincronismo, produzindo resultados mais significativos quando os canais do AND são mais profundos.

Os resultados obtidos para um ambiente com uma distribuição "pesada" são apresentados pelas Tabelas: 6.27, 6.29, 6.28 e 6.30. Um resumo dos resultados pode ser visualizado por meio da Figura 6.21.

A inserção de uma atividade mais "pesada" aumentou o tempo de atraso por caso atrasado, mas manteve a redução significativa apresentada na situação com atividades homogêneas. A semelhança de comportamento ocorre devido ao grande número de filas (maior profundidade do canal), permitindo que o algoritmo corrija a tempo as prioridades de cada caso.

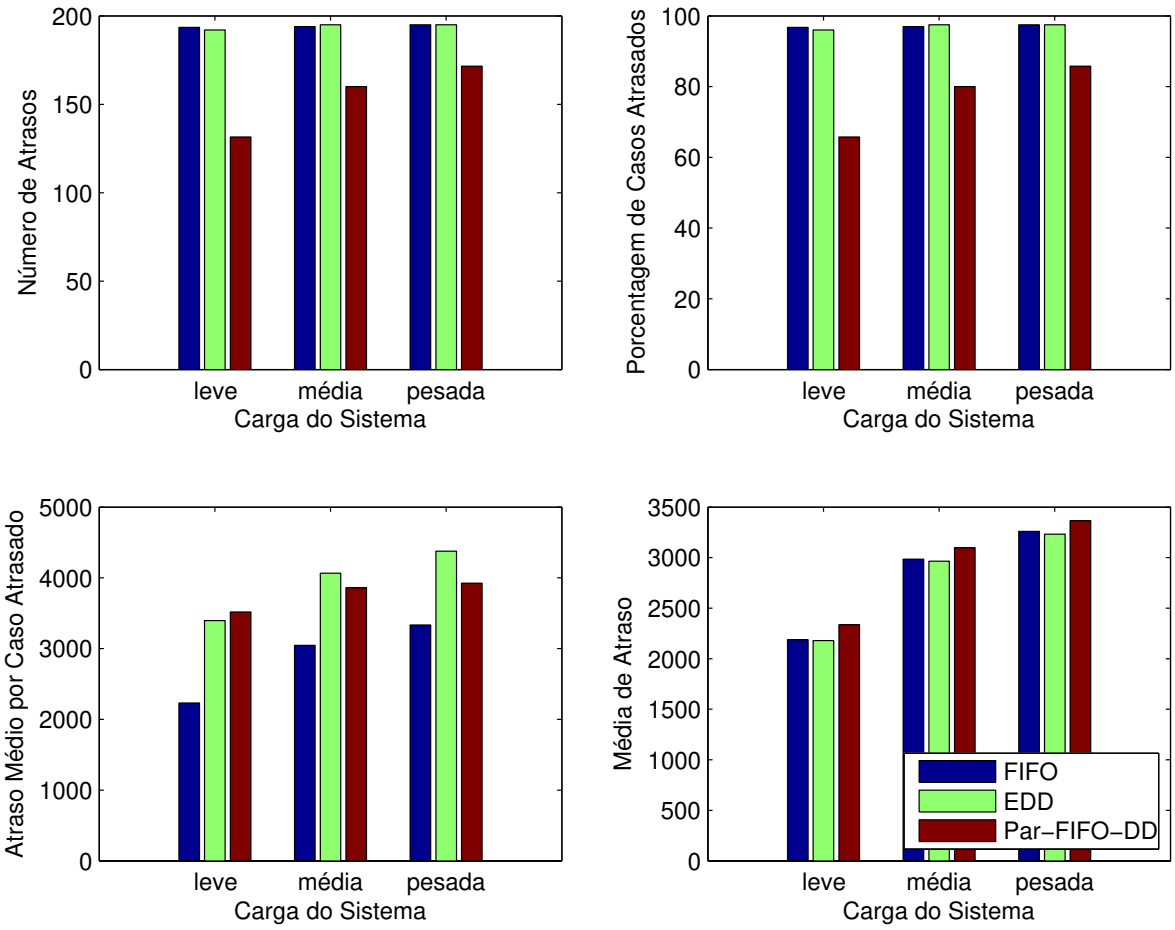


Figura 6.20: Representação gráfica do número de atrasos médios, porcentagem de casos atrasados, atraso médio por caso atrasado e média de atraso. Dados referentes a simulação: cenário - 3; atividades homogêneas; erro máximo de 30%.

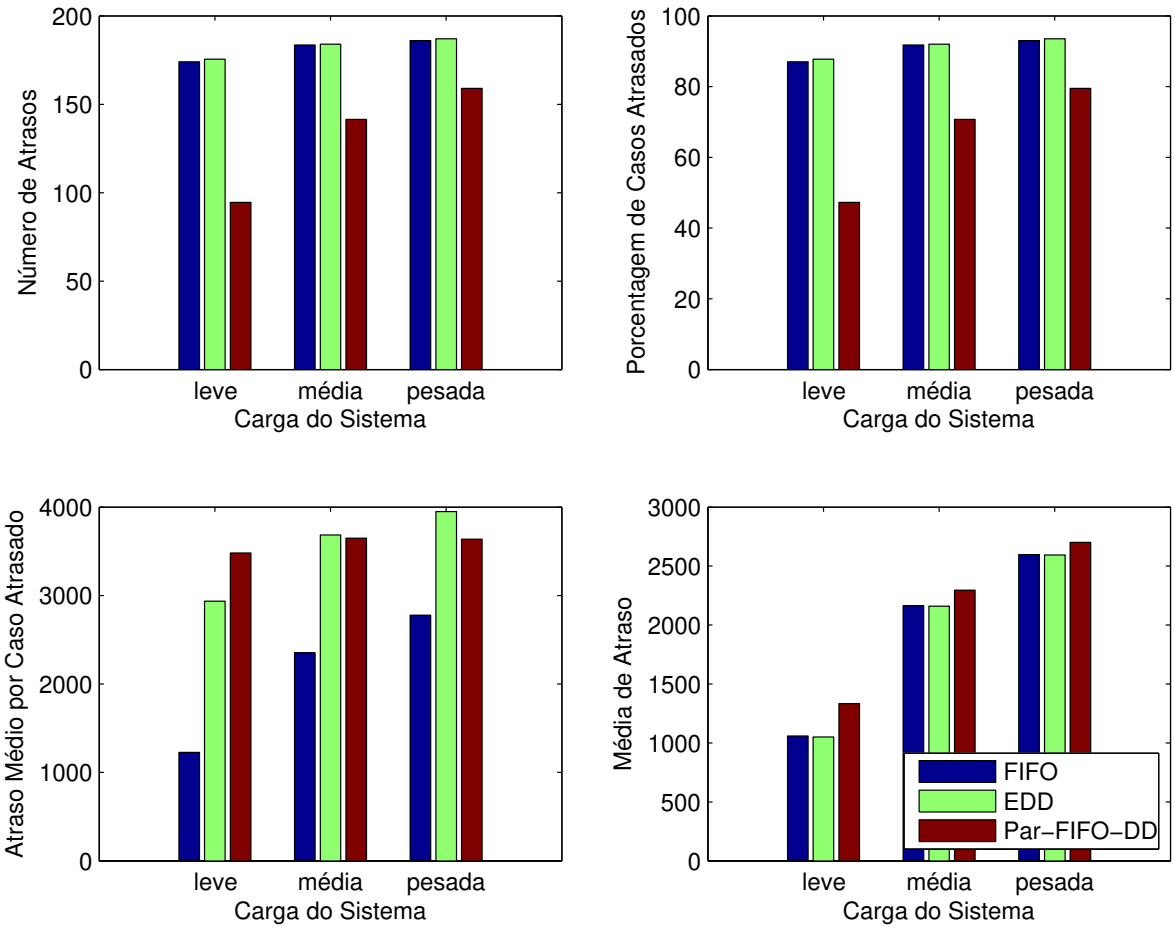


Figura 6.21: Representação gráfica do número de atrasos médios, porcentagem de casos atrasados, atraso médio por caso atrasado e média de atraso. Dados referentes a simulação: cenário - 3; uma atividade com distribuição "pesada"; erro máximo de 30%.

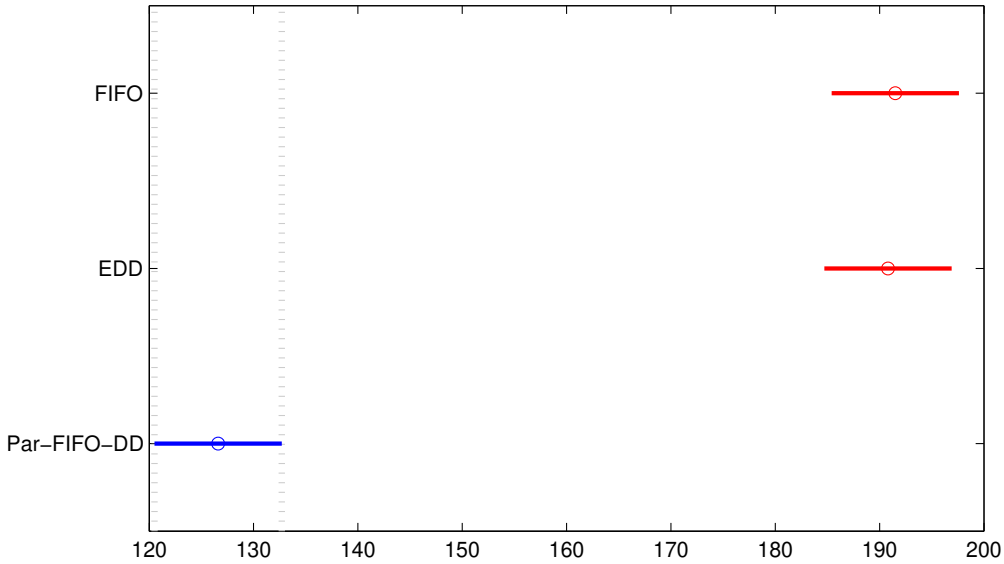


Figura 6.22: Avaliação por meio da ANOVA com Tukey para o número de atrasos: cenário - 3; atividades homogêneas; erro máximo de 30%; sistema com uma carga "pesada".

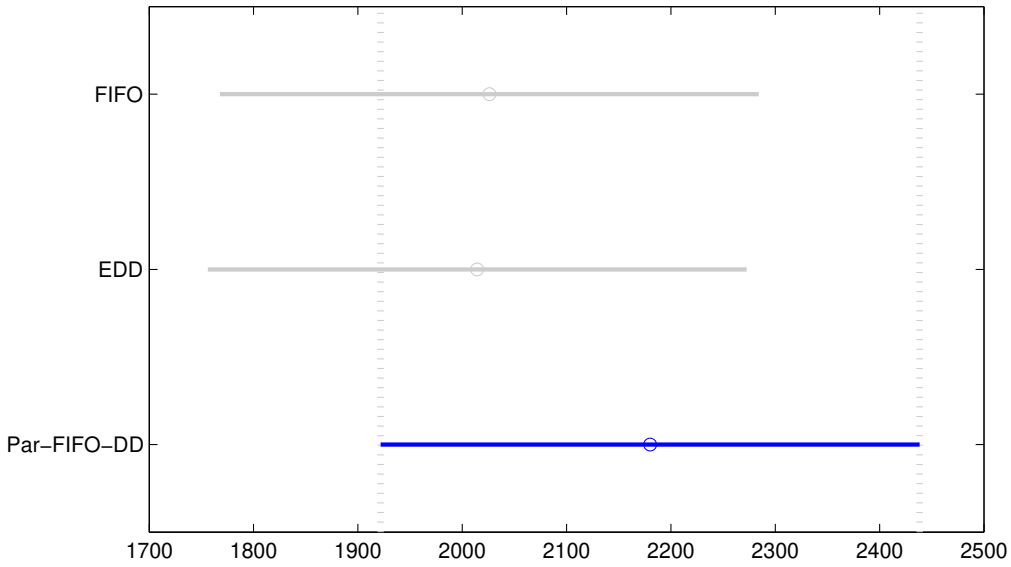


Figura 6.23: Avaliação por meio da ANOVA com Tukey para média de atraso: cenário - 3; atividades homogêneas; erro máximo de 30%; sistema com uma carga "pesada".

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	193,5	194	195
EDD	192	195	195
Par-FIFO-DD(10)	129	160	162
Par-FIFO-DD(30)	131,5	160	171,5
Par-FIFO-DD(50)	133	161	183,5

Tabela 6.23: Resultados: número médio de atrasos; cenário - 3, atividades homogêneas.

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	2187,993	2984,57	3259,68
EDD	2178,998	2964,995	3231,983
Par-FIFO-DD(10)	2446,466	3100,932	3248,754
Par-FIFO-DD(30)	2336,12	3096,82	3364,858
Par-FIFO-DD(50)	2264,548	3089,963	3146,125

Tabela 6.24: Resultados: atraso médio; cenário - 3; atividades homogêneas.

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	2232,645	3045,46	3334,758
EDD	2217,825	3041,86	3292,297
Par-FIFO-DD(10)	3585,328	3890,293	4156,03
Par-FIFO-DD(30)	3518,231	3859,399	3925,574
Par-FIFO-DD(50)	3146,328	3802,33	3742,53

Tabela 6.25: Resultados: atraso médio por casos atrasado; cenário - 3, atividades homogêneas.

Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	96,75	97	97,5
EDD	96	97,5	97,5
Par-FIFO-DD(10)	64,5	80	81
Par-FIFO-DD(30)	65,75	80	85,75
Par-FIFO-DD(50)	66,5	80,5	91,75

Tabela 6.26: Resultados: porcentagem de atrasos; cenário - 3; atividades homogêneas.

Número Médio de Atrasos			
Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	174	183,5	186
EDD	175,5	184	187
Par-FIFO-DD(10)	90	131	148
Par-FIFO-DD(30)	94,5	141,5	159
Par-FIFO-DD(50)	125	153,5	172

Tabela 6.27: Resultados: número médio de atrasos; cenário - 3, uma atividade com distribuição “pesada”.

Atraso Médio			
Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	1058,598	2163,513	2597,443
EDD	1050,64	2159,66	2594,043
Par-FIFO-DD(10)	1553,542	2950,424	2954,42
Par-FIFO-DD(30)	1334,005	2295,683	2701,35
Par-FIFO-DD(50)	1764,557	2490,369	2922,215

Tabela 6.28: Resultados: atraso médio ; cenário - 3, uma atividade com distribuição "pesada".

Atraso Médio Por Caso Atrasado			
Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	1227,423	2352,268	2777,585
EDD	1214,877	2328,217	2774,377
Par-FIFO-DD(10)	4054,129	4688,172	3976,884
Par-FIFO-DD(30)	3481,224	3647,8	3636,231
Par-FIFO-DD(50)	4604,794	3957,154	3933,533

Tabela 6.29: Resultados: atraso médio por caso atrasado; cenário - 3, uma atividade com distribuição "pesada".

Porcentagem de Casos Atrasados			
Técnica	**Carga do Sistema**		
	leve	*médio*	*pesado*
FIFO	87	91,75	93
EDD	87,75	92	93,5
Par-FIFO-DD(10)	45	65,5	74
Par-FIFO-DD(30)	47,25	70,75	79,5
Par-FIFO-DD(50)	62,5	76,75	86

Tabela 6.30: Resultados:porcentagem de atraso; cenário - 3, uma atividade com distribuição “pesada”.

Capítulo 7

Conclusões

Organizar, gerir e utilizar de forma eficiente e coerente os recursos disponíveis são fatores essenciais para o sucesso de uma organização. O controle dos processos de negócio e a redução do tempo necessário para a execução de tarefas, obtidos pela implementação de um sistema de *workflow*, é amplamente demonstrado na literatura de *workflow*. Porém, avaliando as estruturas de roteamento que compõem tais sistemas, possíveis melhorias se tornam aparentes. Nesse sentido, a aplicação de técnicas de escalonamento para ordenar o trabalho contribui para otimizar quesitos valiosos, como o número de instâncias atrasadas ou o tempo que cada instância passa no sistema.

Neste trabalho, as estruturas de roteamento paralelas (AND), amplamente encontradas em sistemas de *workflow*, foram analisadas com o intuito de desenvolver políticas de escalonamento específicas para tais estruturas. Quesitos como a correlação de partes de um mesmo caso e previsões de tempo de processamento foram utilizados no desenvolvimento de três heurísticas de escalonamento, sendo uma delas voltada à redução do número de instâncias atrasadas, e as outras duas com o objetivo de reduzir o tempo médio de processamento.

A comparação dos resultados obtidos pelas heurísticas propostas com as técnicas mais utilizadas em sistemas de *workflow* demonstra que a utilização de políticas de escalonamento específicas para estruturas de roteamento do tipo AND é mais eficiente que a FIFO e EDD na redução do número de instâncias atrasadas para todos os cenários avaliados, apresentando os melhores resultados em cenários semelhantes ao 1 e 2 com cargas leves. Vale ressaltar que a redução do número de atrasos implica em um aumento no tempo de atraso por caso atrasado; assim, caso o objetivo seja reduzir o atraso médio por caso atrasado, recomenda-se a utilização da política FIFO.

Outro resultado importante a ser relatado é a relação entre a redução do tempo médio de processamento e o número de atividades dispostas em série em cada um dos canais do AND. Um maior número de atividades em série permite que o algoritmo de escalona-

mento corrija uma atribuição mal sucedida ou a prioridade dada a uma certa instância, contribuindo para redução do tempo médio de processamento.

A redução no tempo médio de processamento não é tão expressiva para todos os Cenários. Para os cenários 1 e 3, os resultados foram mais promissores, permitindo uma melhor sincronização das instâncias e resultando em uma redução no tempo médio de processamento. Em todas as situações e, apesar da proximidade dos resultados, a heurística PT-Ratio obteve melhores resultados que a Sync-SPT.

Para o Cenário - 1, é recomendado utilizar a PT-Ratio ao invés da SPT em todas as situações de carga; porém, para cargas leves e com atividades com um comportamento homogêneo, FIFO pode ser vista como uma boa alternativa, pois apresenta resultados semelhantes a PT-Ratio e Sync-SPT. Para situações onde existe atividades com um comportamento "pesado", PT-Ratio é mais recomendada.

Para o Cenário - 3 é recomendado utilizar a PT-Ratio ou Sync-SPT em todas as situações, podendo ser observada uma maior diferença entre as políticas FIFO e SPT em cargas mais leves.

Já para o Cenário - 2, com cargas leves e medianas, os resultados da FIFO, SPT, Sync-SPT e PT-Ratio foram semelhantes, havendo uma leve tendência para a SPT. A semelhança desses resultados viabiliza a utilização de políticas mais simples. Porém, para cargas pesadas, as políticas PT-Ratio e Sync-SPT são mais recomendadas. Em situações em que há uma atividade "pesada", o comportamento se manteve.

Os bons resultados obtidos nesta pesquisa estimulam a investigação da políticas de escalonamento especificas para diferentes estruturas de roteamento e para cenários mais complexos.

7.1 Trabalhos Futuros

Dentre os trabalhos vislumbrados destacam-se:

- Neste trabalho, foi identificado que em cenários não balanceados a redução no tempo médio de processamento pela heurística proposta não foi relevante. Esses pontos de desequilíbrio geram os chamados gargalos de sistema. A identificação desses gargalos em cenários mais complexos (compostos por OR, LOOP, AND e Sequência) pode ser utilizada no desenvolvimento de uma técnica de escalonamento que use tais pontos como subsistemas.

- Construir um sistema dinâmico para definir o *allowance factor*, ou seja, modelar a variável A como uma função para que as condições atuais do sistema de *workflow* sejam consideradas na definição de prazos.

Referências Bibliográficas

[1] Rob Allen. *Workflow: An Introdution*. Published in Association with the Workflow Management Coalition (WfMC), Lighthouse Point, FL, USA, layna fischer edition, October 2001.

[2] Jerry Banks, S. John Carson, L. Barry Nelson, and M. David Nicol. *Discrete-Event System Simulation*. Prentice Hall, New Jersey, USA, 5 edition, 2009.

[3] Daniel Barbara, Sharad Mehrotra, and Marek Rusinkiewicz. Incas: A computation model for dynamic workflows in autonomous distributed environments. *Journal of Database Management (JDM)*, 7(1):5 – 15, April 1994.

[4] René Bekker, C. Sem Borst, J. Boxma Onno, and Offer Kella. Queues with workload-dependent arrival and service rates. *Queueing Syst. Theory Appl.*, 46:537–556, March 2004.

[5] Claudio Bettini, Xiaoyang Sean Wang, and Sushil Jajodia. Temporal reasoning in workflow systems. *Distributed and Parallel Databases*, 11(3):269–306, 2002.

[6] Luiz F. Bittencourt, Carlos R. Senna, and Edmundo R. Madeira. Bicriteria service scheduling with dynamic instantiation for workflow execution on grids. In *Proceedings of the 4th International Conference on Advances in Grid and Pervasive Computing*, GPC '09, pages 177–188, Berlin, Heidelberg, 2009. Springer-Verlag.

[7] Jacek Blazewicz, K. Jan Lenstra, and Rinnooy Kan. Scheduling subject to resource constraints: Classification and complexity. *Discrete Applied Mathematics*, 5(1):11 – 24, 1983.

[8] Peter Brucker. *Scheduling Algorithms*. Springer-Verlag Heidelberg, New York, 5 edition, march 2007.

[9] F. Casati, S. Ceri, B Pernici, and G. Pozzi. Conceptual modeling of workflows. In Michael Papazoglou, editor, *OOER '95: Object-Oriented and Entity-Relationship*

Modeling, volume 1021 of *Lecture Notes in Computer Science*, pages 341–354. Springer Berlin / Heidelberg.

[10] Chin Soon Chong, Malcolm Yoke Hean Low, Appa Iyer Sivakumar, and Kheng Leng Gay. Using simulation based approach to improve on the mean cycle time performance of dispatching rules. In *Proceedings of the 37th Conference on Winter Simulation*, WSC '05, pages 2194–2202. Winter Simulation Conference, 2005.

[11] Workflow Management Coalition. Wfmc: Workflow reference model (tc00-1003). WfMC - Online PDF, February 1995. Technical Report.

[12] Workflow Management Coalition. Wfmc: Terminology and glosssary (wfmc-tc-1011). WfMC - Online PDF, February 1999. Glossary.

[13] Carlo Combi and Giuseppe Pozzi. Task scheduling for a temporal workflow management system. In *Proceedings of the Thirteenth International Symposium on Temporal Representation and Reasoning*, TIME '06, 2006.

[14] Richard Walter Conway, William L. Maxwell, and Louis W. Miller. *Theory of Scheduling*. Dover Books on Mathematics. Dover Publications, 2003.

[15] Sofia Mara de Souza. Estendendo ambientes de suporte a trabalho cooperativo com base no conceito de workflow. Master's thesis, IC - Universidade Estadual de Campinas, Abril 2003.

[16] Maciej Drozdowski. Scheduling multiprocessor tasks – an overview. *European Journal of Operational Research*, 94(2):215–230, February 1996.

[17] Johann Eder and Euthimios Panagos. Managing time in workflow systems. In *in Workflow Handbook 2001, Layna Fischer (Ed.), Future Strategies Inc., 2001*, pages 109–132, 2000.

[18] E. M. Goldratt and J. Cox. *The Goal: A Process of Ongoing Improvement*. North River Press, 2 edition, 1992.

[19] R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of Discrete Mathematics*, 5(2):287–326, 1979.

[20] Kees Van Hee, Hajo Reijers, Eric Verbeek, and Loucif Zerguini. On the optimal allocation of resources in stochastic workflow nets. In *17th UK Performance Engineering Workshop*, pages 23 – 34, Leeds, 2001. Services,University of Leeds.

[21] Myles Hollander and Douglas A. Wolfe. *Nonparametric Statistical Methods*. Wiley-Interscience, New York, 2 edition, January 1999.

[22] W. Kelton, Randall Sadowski, and Nancy Swets. *Simulation with Arena*. McGraw-Hill Science, New York, 5rd edition, 2009.

[23] J.P.C. Kleijnen and W. Groenendaal. *Simulation: A Statistical Perspective*. Wiley, 1992.

[24] Manuel Laguna and Johan Marklund. *Business Process Modeling, Simulation and Design*. Prentice Hall, Upper Saddle River, New Jersey, 2004.

[25] Manuel Laguna and John Peter Marklund. *Business Process Modeling, Simulation and Design*. Prentice Hall, 2004.

[26] Stephen R. Lawrence and Edward C. Sewell. Heuristic, optimal, static, and dynamic schedules when processing times are uncertain. *Journal of Operations Management*, 15(1):71 – 82, 1997.

[27] Ronny S. Mans, Nick C. Russell, Wil M. P. van der Aalst, Arnold J. Moleman, and Piet J. M. Bakker. Transactions on petri nets and other models of concurrency iv. chapter Schedule-aware workflow management systems, pages 121–143. Springer-Verlag, Berlin, Heidelberg, 2010.

[28] Manolis Marazakis and Christos Nikolaou. Towards adaptive scheduling of tasks in transactional workflows. In *WSC '95: Proceedings of the 27th Conference on Winter Simulation*, pages 604–611, Washington, DC, USA, 1995. IEEE Computer Society.

[29] D. C. Mattfeld and C. Bierwirth. Minimizing job tardiness: Priority rules vs. adaptive scheduling. In I. C. Parmee, editor, *Adaptive Computing in Design and Manufacture*, pages 59–67. Springer, 1998.

[30] Thomas M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition, 1997.

[31] Michael L. Pinedo. *Scheduling: Theory, Algorithms, and Systems*. Springer Publishing Company, Incorporated, 3rd edition, 2008.

[32] Hajo A. Reijers. *Design and control of workflow processes: business process management for the service industry*. Springer-Verlag, Berlin, Heidelberg, 2003.

[33] Leonardo Hartleben Reinehr. Um sistema de gerenciamento de worklfows para ambientes de comunicaçãoo sem fio. Master's thesis, IC - Universidade Estadual de Campinas, 2002.

[34] Anne Rozinat, Moe Wynn, Wil Aalst, Arthur Hofstede, and Colin Fidge. Workflow simulation for operational decision support using design, historic and state information. In *Proceedings of the 6th International Conference on Business Process Management*, BPM '08, pages 196–211, Berlin, Heidelberg, 2008. Springer-Verlag.

[35] Marek Rusinkiewicz and Amit Sheth. Modern database systems. chapter Specification and execution of transactional workflows, pages 592–620. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 1995.

[36] Ayyub Shahaan and Abramson David. Gridrod: A dynamic runtime scheduler for grid workflows. In Burton J. Smith, editor, *ICS*, pages 43–52. ACM, 2007.

[37] Munindar P. Singh. Distributed scheduling of workflow computations. In *International Conference on Data Engineering (ICDE)*, volume 12, New Orleans, Louisiana, February 1996.

[38] N. Tatbul, S. Nural, P. Karagöz, İ. Çingil, E. Gökkoca, M. Altınel, P. Köksal, A. Doğaç, and T. Özsu. A Workflow Specification Language and its Scheduler. In *International Symposium on Computer and Information Sciences (ISCIS'97)*, Antalya, Turkey, October 1997.

[39] Vincent T'Kindt and Jean Charles Billaut. *Multicriteria Scheduling - Theory Models and Algorithms*, volume 1. Springer, Berlin, 2 edition, 2006.

[40] B. Gregório Tramontina. *Composicionalidade de Técnicas de Escalonamento de Processos em Workflow*. PhD thesis, Instituto de Computação - Universidade Estadual de Campinas, 2008.

[41] B. Gregório Tramontina, Jacques Wainer, and Clarence Ellis. Applying scheduling techniques to minimize the number of late jobs in workflow systems. In *Proceedings of the 2004 ACM Symposium on Applied Computing*, SAC '04, pages 1396–1403, New York, NY, USA, 2004. ACM.

[42] Wil M. P. van der Aalst and Kees van Hee. *Workflow Management: Models, Methods, and Systems*. MIT Press, 2002.

[43] Vijay V. Vazirani. *Approximation Algorithms*. Springer, Berlin, 1 edition, 2001.

[44] Wanpeng Zhang and Jing Chen. Modeling and verification for planning and scheduling in a workflow framework. In *Proceedings of the 2010 International Conference on Intelligent Computation Technology and Automation*, volume 2 of *ICICTA '10*, pages 714–717, Washington, DC, USA, 2010. IEEE Computer Society.

[45] Henan Zhao and Rizos Sakellariou. Scheduling mmltiple dags onto heterogeneous systems. In *Proceedings of the 20th International Conference on Parallel and Distributed Processing*, IPDPS'06, pages 159–159, Washington, DC, USA, 2006. IEEE Computer Society.

[46] J. Leon Zhao and Edward A. Stohr. Temporal workflow management in a claim handling system. *SIGSOFT Softw. Eng. Notes*, 24:187–195, March 1999.