Proceeding Series of the Brazilian Society of Computational and Applied Mathematics, v. 8, n. 1, 2021.

Trabalho apresentado no XL CNMAC, Evento Virtual - Co-organizado pela Universidade do Mato Grosso do Sul (UFMS).

Proceeding Series of the Brazilian Society of Computational and Applied Mathematics

pontoSim: concepção e simulação de *pipelines* para o processamento de nuvens de pontos 3D

Luiz F. F. Leite[1]
Artur R. R. Neto[2]
José M. Soares[3]
George A. P. Thé[4]
DETI/UFC, Fortaleza, CE

Resumo. Nos trabalhos científicos que envolvem processamento de imagens 3D representadas por nuvens de pontos, experimentos e simulações exigem do pesquisador esforços significativos para configuração do ambiente e conhecimentos avançados em programação. Bibliotecas para processamento de nuvens de pontos 3D são complexas e, geralmente, possuem dependências que restringem o uso em alguns dispositivos. Considerando que o tempo disponível para desenvolvimento de projetos científicos é limitado, a preparação das condições necessárias pode impactar significativamente a produtividade. Visando contribuir com redução dos esforços indiretos no desenvolvimento de projetos científicos, foi concebido o arcabouço pontoSim, que permite a concepção de *pipelines* para processamento de nuvens de pontos 3D usando a notação de Redes de Petri Coloridas. O pontoSim oferece suporte gráfico e exige baixo esforço de programação. Para executar simulações, pontoSim é integrado a uma biblioteca desenvolvida em C, que não possui dependências e exige baixo esforço de implantação, sendo portável para diferentes plataformas de *hardware*.

Palavras-chave. Nuvem de Pontos 3D, Modelagem e Simulação, Rede de Petri, Visão Computacional

1 Introdução

Bibliotecas para processamento de imagens 3D, como PCL [12] e Open3D [16], permitem a manipulação de nuvens de pontos, estruturas de dados frequentemente adotada para representação de imagens 3D. Recursos consolidados para o processamento deste tipo de imagens estão disponíveis em tais bibliotecas, oferecendo suporte a pesquisadores para concepção e experimentação de novas técnicas. É preciso, entretanto, conhecimentos avançados em programação, além daqueles necessários à configuração e instalação das bibliotecas no ambiente de desenvolvimento. Isso pode representar um esforço significativo, impactando no tempo e custo das pesquisas, principalmente para pesquisadores que não possuam experiência e produtividade em desenvolvimento de *software*.

Tendo em perspectiva a concepção e simulação rápida de *pipelines* para processamento de imagens 3D representadas por nuvens de pontos, propõe-se neste trabalho um arcabouço baseado em modelagem por Redes de Petri Coloridas (RPC) [6], o pontoSim. Estima-se facilitar a concepção de novas técnicas com a disponibilização transparente dos recursos de uma biblioteca de funções aos usuários. A articulação destes recursos por modelagem em alto nível favorece a implementação de algoritmos sem a necessidade de conhecimentos avançados em programação. Os modelos em

[1]luizleite@alu.ufc.br
[2]artur.rodrigues26@gmail.com
[3]marques@ufc.br
[4]george.the@ufc.br

RPC permitem, ainda, que se tenha uma visão mais didática dos modelos concebidos, podendo auxiliar o estudo e a compreensão dos algoritmos propostos por terceiros. De forma a reduzir a necessidade de configuração de ambientes sofisticados de programação, o arcabouço proposto utiliza uma biblioteca de funções escrita em C denominada Pontu, que foi concebida no seio do grupo de pesquisa no qual o presente trabalho foi desenvolvido. A Pontu possui como características fundamentais independência de outras bibliotecas, facilidade de implantação e alta portabilidade para diferentes dispositivos computacionais.

Trabalhos correlatos ao pontoSim permitem a concepção e simulação de *pipelines* para algoritmos de diversas naturezas, exigindo do pesquisador menor esforço para proposição de soluções computacionais. Fiji [13] é uma plataforma gratuita e de código aberto que permite o desenvolvimento de algoritmos de análise de imagens biológicas. É disponibilizado em forma de *plugin* para o *software* ImageJ [14]. A plataforma cria um ecossistema em que comunidades de ciência da computação e de biologia podem colaborar em problemas cujas soluções requeiram conhecimentos em ambas as áreas. Os recursos do Fiji são explorados por meio de linguagens de *scripts*, proporcionando um meio para construção de *pipelines* avançados de processamento de imagens. Estes, por sua vez, retroalimentam a plataforma e podem ser usados por outros pesquisadores.

Diferentemente do Fiji, o Simulink [15] é um ambiente que permite modelagem e simulação de sistemas de maneira gráfica, sem que sejam necessários conhecimentos avançados em programação. A ferramenta tem alta integração com o MATLAB [8], permitindo utilização de programação em sua linguagem proprietária e acesso a seus milhares de algoritmos já implementados. Pode ser utilizado em múltiplas áreas, desde a prototipação de sistemas para análise e processamento de sinais a aplicações de inteligência artificial. O Simulink é capaz de gerar código automaticamente em C e HDL (*Hardware Description Language*, Linguagem de Descrição de *Hardware*) a partir dos modelos construídos. Os códigos gerados podem ser embarcados diretamente em processadores ou FPGA (*Field Programmable Gate Array*, Arranjo de Portas Programáveis em Campo)/ASIC (*Application Specific Integrated Circuit*, Circuito Integrado de Aplicação Específica). O MATLAB/Simulink dispõe de um módulo que permite a utilização de estruturas e funções voltadas para o processamento de nuvens de pontos 3D [10]. Apesar de ser um ambiente rico, que pode ser aplicado a múltiplos domínios e processar dados em diferentes formatos, o MATLAB e seus módulos, como o Simulink, não são recursos gratuitos e nem de código aberto.

As plataformas Fiji e Simulink possuem valor inquestionável, no entanto estima-se que o arcabouço proposto pode proporcionar grande valor para o desenvolvimento de *pipelines* com destaque para os seguintes diferencias: (**i**) é voltado exclusivamente para processamento de nuvens de pontos 3D; (**ii**) é baseado em uma biblioteca de alta performance, baixa dependência e alta compatibilidade com dispositivos diversos; (**iii**) permite o processamento em dispositivos remotos devido a sua arquitetura cliente-servidor com comunicação via protocolo TCP; (**iv**) utiliza RPC, uma ferramenta matemática de modelagem já conhecida na comunidade científica, que conta com recursos para análise de seus modelos; e (**v**) é uma ferramenta gratuita e de código aberto.

2 pontoSim

O arcabouço pontoSim se apoia em uma arquitetura cliente-servidor. A organização dos componentes e seus módulos é apresentada na Figura 1.

O componente cliente é composto pela ferramenta de modelagem em RPC e pelos módulos responsáveis pela comunicação com componente servidor. O **Editor de RPC** implementa a interface gráfica para construção, análise e simulação dos modelos em RPC e, na versão atual do pontoSim, é constituído pelo CPNTools [11] e um conjunto de declarações em sua linguagem, o CPNML. O **Protocolo pontoSim** é o módulo que implementa o protocolo de requisição de comandos ao servidor. O componente cliente é completado pela **Interface de comunicação**,

DOI: 10.5540/03.2021.008.01.0460

que utiliza o protocolo TCP para troca de mensagens com o servidor. Na versão atual, esta comunicação é feita pelo módulo Comms/CPN [5].

O componente servidor é implementado em C e integrado à biblioteca Pontu para o processamento de nuvens de pontos, contando com alguns módulos com funções específicas. A **Interface de comunicação** é responsável pela troca de mensagens com o módulo homônimo no cliente, gerenciando requisições e respostas, e foi escrito com base no código disponível pelo módulo Comms/CPN [3]. O módulo **Gerenciador de sessões** implementa o conceito de sessão utilizado no arcabouço. O módulo **Comandos** implementa a integração com as funções da biblioteca que estão disponíveis para configuração dos blocos do *pipeline* modelados pelo usuário. O módulo **Processamento de nuvem de pontos 3D** representa a biblioteca que fornece as funções para o processamento de nuvens de pontos 3D, cuja integração é configurada no bloco **Comandos** para cada função. Este módulo do pontoSim é implementado pela biblioteca Pontu. É importante frisar o caráter modular da arquitetura proposta e de seus componentes, o que a torna adaptável para uso de outras bibliotecas de processamento de imagem.

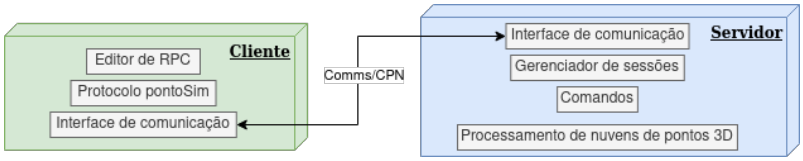


Figura 1: Arquitetura de pontoSim. Em verde o componente cliente, em azul o componente servidor, e em cinza os módulos de cada componente.

O pontoSim implementa cinco tipos distintos: números inteiros, números reais, texto, nuvens de pontos e matrizes. Os dois últimos tipos são tratados como objetos. Todas as mensagens são enviadas pelo cliente ao servidor no formato texto. O servidor, ao receber a mensagem, converte-a para o formato esperado pelo comando, que deve ser utilizado para manipular o dado durante a execução dele. De maneira análoga, o servidor converte suas respostas para texto antes de enviar suas mensagens para o cliente. As mensagens recebidas no cliente, por sua vez, devem ser convertidas explicitamente do formato texto para o formato apropriado, caso seja necessário [5].

Nuvens de pontos e matrizes mantidas em memória juntamente aos seus identificadores únicos são tratados no arcabouço como objetos [6]. Cada objeto pertence a uma sessão. As sessões identificam a que cliente pertence uma lista objetos de uma simulação. Os objetos podem ser excluídos em duas situações: (**i**) após a sua utilização como parâmetro de entrada para um comando; e (**ii**) no fim de uma sessão, normalmente definida no final de uma simulação.

Após a execução de um comando, por padrão, o servidor exclui os objetos que foram utilizados como parâmetros de entrada para o comando. Entretanto, os objetos podem ser mantidos e, para essa finalidade, podem ser utilizadas funções que definem esse comportamento. Uma sessão iniciada por um cliente pode ser encerrada também pelo próprio cliente. Sessões podem ser encerradas automaticamente pelo servidor quando alcançam um tempo limite, evitando assim que uma simulação aloque recursos por tempo indeterminado, caso a simulação falhe, não seja concluída, ou o usuário não defina o fim da sessão explicitamente.

[5] A proposta trazida pelo arcabouço é de executar o processamento dos dados no servidor. Os dados são, portanto, mantidos em formato de texto no componente cliente, agilizando a troca de mensagens.

[6] As nuvens de pontos e matrizes utilizadas nas rotinas do arcabouço são, em geral, volumosas. A troca de mensagem entre cliente e servidor incluindo esses tipos de dados causaria sobrecarga desnecessária no lado do cliente, já que todo processamento desses dados é feito no servidor. Por isso, estruturas dessa natureza são mantidas na memória do servidor, sendo incluídos nas mensagens apenas identificadores desses dados do servidor para o cliente.

DOI: 10.5540/03.2021.008.01.0460

Exemplos de uso do arcabouço, lista de comandos e instruções sobre como criar comandos podem ser encontrados no repositório do projeto [7].

3 Biblioteca Pontu

A biblioteca Pontu, escrita na linguagem de programação C, oferece recursos para análise e manipulação de nuvens de pontos 3D. Como característica de fundamental importância para uso no arcabouço proposto, a biblioteca não possui dependência de recursos de terceiros. Utiliza apenas a biblioteca padrão da linguagem C e é compatível com o padrão C11, sendo portável para diferentes plataformas de *hardware*.

A Pontu se divide em quatro módulos que implementam, além de estruturas de dados básicas e funções para manipulação de nuvens de pontos 3D, extratores de características, algoritmos de alinhamento e métodos de subamostragem.

Na arquitetura do pontoSim, a Pontu é utilizada pelo componente servidor. Neste componente, as requisições incluem um comando relativo a uma função da Pontu, além de uma lista de parâmetros que depende das necessidades específicas ao processamento requisitado para uma ou mais nuvens de pontos 3D.

A vantagem fundamental de usar a biblioteca Pontu no arcabouço proposto em relação a outras bibliotecas comumente usadas para o processamento 3D é a ausência de dependências complementares, além do seu tamanho reduzido. Estes fatos vem ao encontro das motivações apresentadas para este trabalho, visto que, quanto maior o nível de dependência de um *software* ou componente, mais será exigido em termos de configuração e, consequentemente, em conhecimentos técnicos específicos para simulações. Adicionalmente, este componente pode ser implantado em dispositivos distintos, incluindo aqueles com processadores de arquitetura ARM, viabilizando seu uso até mesmo em *smartphones*.

Pontu está em constante desenvolvimento. Acesso a biblioteca e adição de funcionalidades a ela podem ser requisitados, bastando que se faça contato com seus autores.

4 Modelagem por RPC

As RPC, utilizadas neste trabalho para dar suporte à modelagem em alto nível dos *pipelines* de processamento de nuvens de pontos 3D, são uma extensão das Rede de Petri (RP) [9], uma ferramenta gráfica e matemática inicialmente proposta para modelagem de sistemas dinâmicos a eventos discretos. As RP possuem três elementos básicos [2]: lugar, transição e ficha.

O lugar pode ser interpretado como uma variável de estado do modelo e é representado graficamente por elipses. A transição é associada aos eventos do sistema modelado, representada graficamente por retângulos. Fichas indicam o estado de cada lugar em um momento da execução de um sistema dinâmico, sendo cada ficha representada graficamente por um ponto inserido em um lugar. Assim, as marcações das RP consistem na distribuição de fichas pelos lugares, indicando o estado do sistema em um determinado momento. Os lugares são conectados às transições por arcos direcionados. Estes representam, a depender da orientação, a relação entre uma transição e um lugar de entrada ou de saída. As redes de Petri podem simular a ocorrência de eventos de um sistema com o disparo de transições, que removem fichas dos lugares de entrada e depositam fichas nos lugares de saída, respeitando os pesos dos arcos.

As RPC estendem a representação lugar-transição das RP por meio de uma linguagem de programação de alto nível e definição de fichas mais complexas, que podem possuir valores e tipos distintos [6]. Um valor associado a uma ficha em RPC é chamado de "cor" e tipos de dados possíveis

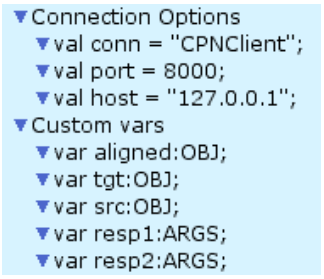
[7] Repositório disponível em: https://gitlab.com/leitelf/pontosim.

em um lugar de "conjunto de cores". As transições ganham novos recursos: guardas e funções. Uma guarda permite a definição de condições que devem ser atendidas para o disparo de uma transição, além daquelas definidas para as RP tradicionais. Uma função adiciona a possibilidade de programação de um comportamento especializado que é executado com o disparo de uma transição. Com a extensão das RPC os arcos podem receber inscrições. As inscrições permitem definição de lógicas específicas para a remoção e adição de fichas aos lugares aos quais eles estão ligados. Para a programação de guardas, funções e inscrições, o CPNTools fornece a linguagem CPNML.

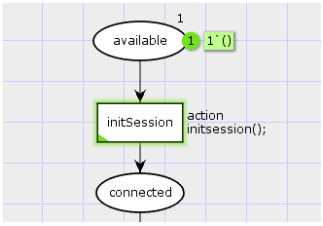
Para modelagem dos *pipelines* no arcabouço, foi utilizada a ferramenta CPNTools. Sua integração aos demais componentes da arquitetura é realizada por meio de *sockets* e do protocolo TCP com auxílio do módulo Comms/CPN [5]. Informações sobre o uso do *software* CPNTools e a linguagem funcional CPNML estão disponíveis no portal da ferramenta [4].

5 Uso do pontoSim para registro de nuvens de pontos 3D

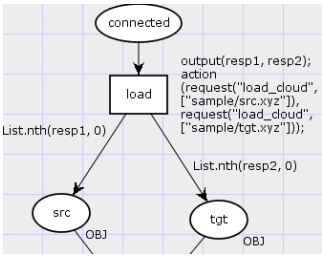
Para exemplificar o uso do pontoSim, foi utilizado o algoritmo *Iterative Closest Point* (ICP) [1] implementado para Pontu. O ICP é um algoritmo iterativo para registro, ou alinhamento, de nuvens de pontos. O registro ocorre quando as regiões em comum de diferentes nuvens de pontos são superpostas, usando-se uma das nuvens como referência, ou alvo.



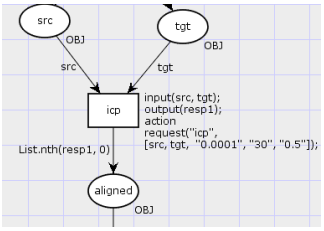
(a) Declaração de variáveis e configurações de conexão.



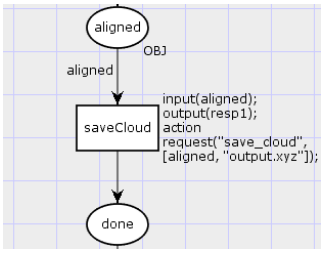
(b) Inicialização de sessão.



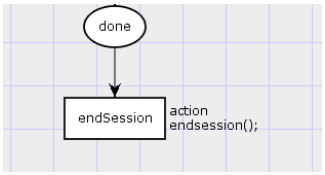
(c) Leitura de arquivos de nuvens de entrada (src) e alvo (tgt).



(d) Alinhamento de nuvem de entrada com alvo por ICP.



(e) Escrita de nuvem resultante em arquivo.



(f) Encerramento de sessão e término da rede.

Figura 2: Etapas de construção de RPC para alinhamento de nuvens de pontos 3D com ICP.

O *pipeline* para alinhar com o ICP uma nuvem de entrada a uma nuvem alvo foi modelado em RPC considerando as seis etapas apresentadas na Figura 2, que podem ser generalizadas para a construção de outros modelos. Utilizando-se as nuvens do coelho de *Stanford* [7], capturadas a 45 graus e 0 graus como nuvens de entrada e alvo, respectivamente, obtém-se o resultado de registro

apresentado na Figura 3 [8].

(a) Vista frontal de nuvem de entrada e nuvem alvo.

(b) Vista frontal de nuvem resultante e nuvem alvo.

Figura 3: Resultado de registro com ICP em nuvens de coelho de *Stanford*. Em verde nuvem alvo, vermelha nuvem de entrada e azul nuvem resultante.

6 Conclusões

Como contribuições deste trabalho, são apresentados o arcabouço pontoSim, para modelagem de *pipelines*, e a Pontu, biblioteca com alta portabilidade desenvolvida em C, no padrão C11. O arcabouço pontoSim, que integra a Pontu, oferece recursos especialmente desenvolvidos para o processamento de nuvens de pontos 3D. Pretende-se com estas contribuições, por um lado, reduzir os esforços necessários à preparação do ambiente para execução de simulações. Por outro lado, visa-se auferir maior produtividade aos ensaios realizados por pesquisadores que não possuam conhecimentos avançados em programação.

O pontoSim permite a concepção e a simulação de algoritmos para processamento de nuvens de pontos 3D por meio de modelagem em RPC. Além da interface gráfica, as RPCs oferecem recursos para análise dos modelos, facilitando a detecção de pontos de paralelismo, impasses e laços infinitos.

A biblioteca Pontu se destaca, quando comparada a outras bibliotecas do segmento, por ser autocontida, livre e expansível. Por não possuir dependência de terceiros, pode ser implantada em vasta gama de dispositivos, tendo como único requisito para instalação e integração de novas contribuições a compatibilidade com a linguagem de programação C, padrão C11.

Atualmente, a principal limitação do arcabouço é a dependência do software CPNTools. Embora o seu uso tenha permitido validar o arcabouço, CPNTools não matém suporte para ambientes Linux e Mac, restrição que vem de encontro aos propósitos do pontoSim e da Pontu. Assim, para a próxima fase de desenvolvimento, prospecta-se uma interface gráfica utilizável a partir de navegadores *web*, com recursos de modelagem em RPC especializados aos requisitos do pontoSim e totalmente independentes do CPNTools. Na nova arquitetura, Pontu poderá ser utilizada remotamente, reduzindo ainda mais as necessidades de preparação do ambiente de produção. Ainda como futuras contribuições, espera-se: a expansão de módulos e funções da biblioteca Pontu; a inclusão de novos comandos para o arcabouço pontoSim; e a implementação de um visualizador *web* para configuração de dados de entrada e avaliação dos resultados de simulações.

Agradecimentos

À Fundação Cearense de Apoio ao Desenvolvimento Científico e Tecnológico (FUNCAP) pela concessão de bolsas. O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

[8] Um vídeo demonstrativo da elaboração desta RPC está disponível em: https://gitlab.com/leitelf/pontosim/-/blob/master/assets/vid/pontosimpres.mp4?expanded=true&viewer=rich.

Referências

[1] Besl, P. J. and McKay, N. D. A method for registration of 3-D shapes, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 14, pages 239-256, 1992. DOI:10.1109/34.121791.

[2] Cardoso, J. e Valette, R. *Redes de Petri*. Editora da UFSC, Florianópolis, 1997.

[3] Comms/CPN. CPN Tools, 2010. Disponível em: <http://cpntools.org/2018/01/09/comms-cpn/>. Acesso em: 05 de Outubro de 2020.

[4] Documentation. CPN Tools, 2010. Disponível em: <http://cpntools.org/2018/01/16/documentation-2/>. Acesso em: 05 de Outubro de 2020.

[5] Gallasch, G. E. and Kristensen, L. M. COMMS/CPN: A communication infrastructure for external communication with Design/CPN, *Third Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools*, DAIMI Report Series, volume PB-554, pages 75-91, 2001.

[6] Jensen, K. and Kristensen, L. M. *Coloured Petri Nets: Modeling and Validation of Concurrent Systems*. Springer Science & Business Media, Berlin, 2009.

[7] Levoy, M., Gerth, J., and Curless, B. The Stanford 3D scanning repository. Stanford Computer Graphics Laboratory, 2014. Disponível em: <http://graphics.stanford.edu/data/3Dscanrep/>. Acesso em: 07 de Julho de 2020.

[8] MATLAB. Mathworks, 2020. Disponível em: <https://www.mathworks.com/products/matlab.html>. Acesso em: 10 de Dezembro de 2020.

[9] Murata, T. Petri nets: Properties, analysis and applications, *Proceedings of the IEEE*, volume 77, pages 541-580, 1989. DOI:10.1109/5.24143.

[10] Point Cloud. Mathworks, 2020. Disponível em: <https://www.mathworks.com/discovery/point-cloud.html>. Acesso em: 10 de Dezembro de 2020.

[11] Ratzer, A. V., Wells, L., Lassen, H. M., Laursen, M., Qvortrup, J. F., Stissing, M. S., Westergaard, M., Christensen, S. and Jensen, K. CPN Tools for Editing, Simulating, and Analysing Coloured Petri Nets, *Applications and Theory of Petri Nets 2003*, Lecture Notes in Computer Science, volume 2679, pages 450-462, 2003. DOI:10.1007/3-540-44919-1_28.

[12] Rusu, R. B. and Cousins, S. 3d is here: Point cloud library (pcl), *2011 IEEE International Conference on Robotics and Automation*, pages 1-4, 2011. DOI:10.1109/ICRA.2011.5980567.

[13] Schindelin, J., Arganda-Carreras, I., Frise, E., Kaynig, V., Longair, M., Pietzsch, T., Preibisch, S., Rueden, C., Saalfeld, S., Schmid, B., Tinevez, J. Y., White, D. J., Hartenstein, V., Eliceiri, K., Tomancak, P. and Cardona, A. Fiji: an open-source platform for biological-image analysis, *Nature methods*, volume 9, pages 676-682, 2012. DOI:10.1038/nmeth.2019.

[14] Schindelin, J., Rueden, C. T., Hiner, M. C. and Eliceiri, K. W. The ImageJ ecosystem: An open platform for biomedical image analysis, *Molecular reproduction and development*, volume 82, pages 518-529, 2015. DOI: 10.1002/mrd.22489.

[15] Simulink. Mathworks, 2020. Disponível em: <https://www.mathworks.com/products/simulink.html>. Acesso em: 10 de Dezembro de 2020.

[16] Zhou, Q. Y., Park, J., and Koltun, V. Open3D: A Modern Library for 3D Data Processing, *arXiv preprint*, 2018. arXiv: 1801.09847.

