情報処理学会論文誌 Vol.58 No.2 396–408 (Feb. 2017)

スマートスペースにおける最小コストでのコンテキスト遷移を可能にするデバイス操作系列導出ツール

水本 旭洋[1,a)] 宵 憲治[1,b)] カレッド エル=ファキ[2,c)] 安本 慶一[1,d)]

受付日 2016年5月23日, 採録日 2016年11月1日

概要：本論文では，スマートスペースにおいて省エネルギーに様々なデバイスをコンテキストアウェア制御することを目的に，任意の 2 つのコンテキスト間を最小コストで遷移させるデバイス操作系列を導出するツール *PathSim* を提案する．PathSim は，スマートスペースにおけるコンテキストの各構成要素の値域を有限個の範囲に分割することで，コンテキストを有限個の要素からなる空間として扱う．そして，各コンテキストにおいて任意のデバイスの操作イベントを実行したときに遷移するコンテキストをシミュレーションにより求めていくことで，最小コストで目的のコンテキストに遷移するデバイスの操作系列を A*アルゴリズムに基づいて導出する．デバイスの状態，部屋の温度/湿度，ユーザの位置情報などをスマートスペースのコンテキストとして扱い，コンテキスト遷移やデバイスの消費電力量をシミュレーション可能なシステムを実装した．日本の夏と冬を想定した計算機シミュレーションを通して評価実験を行った結果，PathSim によって導出された操作系列が，空調機器の制御で一般的に使用されているハイブリッドダイナミカルシステムで動作させる場合と比べ，デバイスの合計消費電力量を最大で約 50%削減できることを確認した．

キーワード：コンテキストモデリング，最短経路問題，ホームオートメーション，コンテキストアウェアネス，スマートスペース

A Tool for Finding Energy-saving Device Operation Sequence for Moving Context in a Smart Space

Teruhiro Mizumoto[1,a)] Kenji Yoi[1,b)] Khaled El-Fakih[2,c)] Keiichi Yasumoto[1,d)]

Received: May 23, 2016, Accepted: November 1, 2016

Abstract: In this paper, aiming to achieve minimal energy consumption in context-aware device control in a smart-space, we propose a tool called *PathSim* which derives a minimal cost sequence of device operations for moving between two arbitrary contexts. PathSim properly divides the domain of each constituent smart-space context into a finite number of ranges, so that a finite discrete context space (finite number of contexts) is obtained. PathSim simulates the context move (range movement of a context variable) by every available event (device operation) at current context, and derives a minimal cost path (event sequence) using the proposed cost functions for events and derived contexts, from the current context into the intended one. To find a minimal cost path, PathSim adopts A* algorithm. PathSim prototype that we implemented can simulate the context moves for the events such as device attributes, temperature, humidity and user location, taking into account the size of the space, performance and power consumption of each device, and so on. Through a computer simulation assuming winter and summer in Japan, we confirmed that PathSim's device operations consume about 50% less energy than a general hybrid dynamical system.

Keywords: context modeling, shortest path problem, home automation, context awareness, smart-space

[1] 奈良先端科学技術大学院大学情報科学研究科
Graduate School of Information Science, Nara Institute of Science and Technology, Ikoma, Nara 630–0192, Japan
[2] シャルジャアメリカン大学
Department of Computer Science and Engineering, American University of Sharjah
[a)] teruhiro-m@is.naist.jp
[b)] yoi.kenji.yc6@is.naist.jp
[c)] kelfakih@aus.edu
[d)] yasumoto@is.naist.jp

© 2017 Information Processing Society of Japan

1. はじめに

ユビキタスコンピューティングシステム [1] やコンテキストアウェアシステム [2] は，周囲の環境をセンシングし，それらに適応するように動作するシステムである．近年，ユビキタスコンピューティングシステムを実現する多くの研究 [3] が行われている．ユビキタスコンピューティングシステムが実装された空間や家はスマートスペース（またはスマートホーム）[4] と呼ばれている．スマートスペースには，センサデバイスや家電デバイスなどが分散して設置されている．センサデバイスはスマートスペースを構成する各空間の温度や湿度といった空間の物理的特性，ユーザの位置情報や行動などの状況をセンシングする．また各デバイスは，スマートスペース内のユーザにサービスを提供するためにネットワークを通して操作される．これらの物理的特性，ユーザの状況，そして家電デバイスの状態などのそれぞれを変数として表したとき，これら変数の値の組はスマートスペースの**コンテキスト**と呼ばれる．スマートスペースは，現在のコンテキストから，ユーザの望むコンテキストに到達するようにデバイスを操作する必要がある．一般的に，複数のデバイスがスマートスペース内に設置されている場合，あるコンテキストから異なるコンテキストにスマートスペースを遷移させるデバイスの操作系列は多数存在する．そして，操作系列ごとに操作するデバイスや操作する順番などが異なるため，消費電力量や動作時間がそれぞれ異なる．たとえば，温度 15°C と湿度 30%（15°C，30%）のコンテキストから，（24°C，50%）のような異なるコンテキストに遷移させる場合を考える．このとき，スマートスペース内に，エアコン，オイルヒータ，そして加湿器が設置されていると，温度を上げるために，エアコンやオイルヒータが使用できるため，エアコンのみを使用する場合や，オイルヒータのみを使用する場合，また両方使用する場合といった異なる操作系列が考えられる．また，デバイスを制御する順番や動作時間によっても異なる操作系列が存在する．実際に，スマートスペースには多くのデバイスが設置されるため，膨大な数の操作系列が存在しうる．そのため，この中から最適または準最適なコストの操作系列を自動的に見つけることができれば，ユーザにとって非常に有益だと考えられる．

本論文では，省エネルギーにコンテキストアウェアなデバイス操作を実現することを目的とし，スマートスペースにおいて，あるコンテキストから任意の別のコンテキストへ遷移する最小コストのデバイスの操作系列を導出するツール *PathSim* を提案する．

PathSim は，A*アルゴリズムを基としたアルゴリズムにより最小コストのデバイス操作系列を導出する．A*アルゴリズムは有限個のノードとコストが付されたエッジで構成されたグラフに対してのみ適用可能なアルゴリズムである．スマートスペースのコンテキストの遷移は，各コンテキストをノードの集合 V，コンテキスト間の遷移をエッジの集合 E とする有向グラフとして表すことができる．しかしながら，一般的にコンテキストの各変数がとりうる値の範囲は連続または無限であるため，そのままでは A*アルゴリズムを適用することができない．そこで，A*アルゴリズムを適用できるように，コンテキストを構成する各属性（変数）の値域を有限個の範囲に分割し，離散的なコンテキスト遷移モデルとして表す．また，A*アルゴリズムが適用されるグラフでは，ノード間を結ぶエッジ（およびコスト）の集合があらかじめ与えられる必要があるが，本論文で対象とするコンテキストの遷移では，エッジの存在およびエッジのコストが未知である．そのため，探索を行う各コンテキストにおいて，隣接するコンテキストへのエッジの存在およびそのコストを動的に調べる必要がある．PathSim は，各コンテキストにおいて実行可能なデバイスの操作（**イベント**と呼ぶ）をシミュレーションにより模擬実行することで，イベントによって遷移する隣接コンテキストと遷移に必要なコストを導出する．そして，A*アルゴリズムに基づきコンテキストの探索を繰り返すことで最小コストで目的のコンテキストに到達するパス（操作系列）を導出する．

日本の冬と夏を想定した計算機シミュレーションにより PathSim の有用性を評価した結果，PathSim で導出した操作系列は，空調機器の制御で一般的に使用されているハイブリッドダイナミカルシステムや消費電力あたりのデバイス能力が最も高いデバイスを温度と湿度それぞれに対して 1 つ選択して制御する手法と比べ，消費電力量を最大で約 50%削減できることを確認した．

2. 関連研究

近年，スマートホーム [5] とコンテキストアウェアシステムに関する様々な研究 [3], [6], [7] が行われている．MavHome [5] では，家の状態（コンテキスト）を感知し，自動的に快適なサービスをユーザに提供する知的エージェントの実現を目的に，住民の行動予測に基づいたコンテキストアウェアサービスを提案している．しかしながら，サービスを提供する際に，エネルギー効率の良いデバイス制御を考慮していない．スマートスペースにおけるコンテキストアウェアサービスはしばしばルール（トリガと動作のペア）の集合で指定される [9]．CADEL [6] は，UPnP プロトコルを利用したコンテキストアウェアなデバイス操作を実現するルールベースのミドルウェアである．また，Gu ら [7] は，オントロジに基づいたコンテキストモデルを定義し，異なる粒度でコンテキストを扱うことができるサービス指向型コンテキストアウェアミドルウェア（SOCAM）を提案している．これらの既存ミドルウェアは，ルールの集合でスマートホームに関する要求仕様を指定しているが，デバイスの制御処理を明示的に指定する必要がある．

© 2017 Information Processing Society of Japan

スマートスペースのデバイスを操作することで，消費エネルギーの最適化を目指すような研究がいくつか行われている．Jahn ら [8] は，デバイスごとの消費電力と累積コスト（エネルギー消費量）を，テレビや携帯電話の画面上にグラフィカルユーザインタフェースを介して，リアルタイムにユーザに提示するエネルギーアウェアなスマートホームを提案している．この研究では，電気エネルギーの動的価格設定をサポートしており，電気料金がしきい値を下回った場合にのみ，指定されたデバイスが起動する機能を提供している．しかし，この研究では，複数デバイスの協調制御をサポートしていない．Kashimoto ら [11] は，省エネのために期間ごとに利用可能なエネルギー量が制限されているという仮定のもと，各家電のエネルギー消費量，ユーザの状況，ユーザの嗜好などを考慮して，居住者の快適さのレベルが最大化されるように，スマートホーム内の家電に利用可能なエネルギー量を準最適配分する手法を提案している．しかしながら，この研究においても，各環境変数の値を変えることができるデバイスは 1 つだけと仮定しており，複数のデバイスを協調させながら，省エネルギーに特定のコンテキストに遷移することは考えていない．

本研究は，スマートホームの管理者がデバイスの詳細な制御処理を指定することなく，省エネルギーに，かつ，複数デバイスを協調制御しながら，スマートスペースを特定のコンテキストに遷移することが可能なデバイス操作系列を導出するツールを実現するという点で既存研究とは大いに異なる．また，本研究では，コンテキストアウェアシステムの制御手法を提案することだけを目的としておらず，コンテキストアウェアシステムの動作テストなど，スマートスペースの実環境を遷移する必要がある様々な用途で利用可能な汎用性の高いツールを実現することを目的としている点で既存研究とは異なる．

3. スマートスペースにおける最小コストデバイス操作問題

本章では，想定するスマートスペースとコンテキストの定義を行う．そして，任意のコンテキストから特定のコンテキストへスマートスペースを遷移させるための最小コストのデバイス操作系列を導出する問題を定義する．

3.1 スマートスペース

3.1.1 コンテキスト

スマートスペースは 1 つまたは複数の（有限個の）物理的な位置（または部屋）の集合 $R = \{r_1, \ldots, r_k\}$ を持つとする．スマートスペースには，テレビ，エアコン，加湿器など様々なデバイスが設置されている．これらのデバイスの集合を，$D = \{d_1, \ldots, d_m\}$ と表す．各デバイス d_i は，それぞれ固有の属性 $A^{d_i}(j)$, $j = 1, \ldots, n$ を持つ．たとえば，テレビの場合，電源状態（Status），チャネル（Current-Channel），音量（Volume-Level）などの属性を持ち，それぞれ $A^{TV}(1)$ = Status, $A^{TV}(2)$ = Current-Channel, and $A^{TV}(3)$ = Volume-Level のように表される．また，デバイスの属性 $A^{d_i}(j)$ は，それぞれ，定義域 $Dom(A^{d_i}(j))$ を持つ．たとえば，テレビの属性は，$Dom(A^{TV}(1)) = \{ON, OFF\}$, $Dom(A^{TV}(2)) = \{1, \ldots, 300\}$, and $Dom(A^{TV}(3)) = \{1, \ldots, 100\}$ のような定義域をそれぞれ持つ．デバイス d_i の現在の状態 c は，デバイスの属性値の組 $v_c^{d_i} = \langle v_c^{d_i}(1), .., v_c^{d_i}(n)\rangle$ $(v_c^{d_i}(j) \in Dom(A^{d_i}(j)), j = 1, \ldots, n)$ で表される．たとえば，テレビの状態は $v_c^{TV} = \langle v^{TV}(1) = \text{"}ON\text{"}, v^{TV}(2) = 10, v^{TV}(3) = 30\rangle$ のように表すことができる．この例では，テレビの電源状態が “ON”，チャネルが 10ch，音量が 30 である．スマートスペース内に設置されている m 個のデバイスの現在の状態は，$\langle v_c^{d_1}, .., v_c^{d_m}\rangle$ で表される．

加えて，スマートスペースに l 人のユーザ $u_1, \ldots, u_l$ がいるものとする．スマートスペース内のユーザの位置は，部屋単位または空間単位で表す．すなわち，各ユーザ u_i は，R の要素のいずれかを $u_i.pos$ に保持する．また，ユーザの行動などの他の属性は $u_i.attr$ として保持されるとする．ユーザ u_i がスマートスペース内にいない場合は，$u_i.pos$ と $u_i.attr$ は，$null$ である．ユーザ u_i の現在の状態も，$v_c^{u_i} = \langle u_i.pos, u_i.attr\rangle$ のように属性値の組として表し，l 人のユーザに関する状態を，$\langle v_c^{u_1}, \ldots, v_c^{u_l}\rangle$ と表す．

さらに，スマートスペースは，各位置 $r_1, \ldots, r_k$ ごとに環境に関する属性を持つものとする．これらの属性として，温度，湿度，照度，騒音レベルなどが考えられるが，本論文では，簡単化のため，温度と湿度のみを扱い，位置 r_i の温度を $r_i.temp$，湿度を $r_i.humidity$ としてそれぞれ表す．これらの属性の値は，デバイスの動作を通して変化し，センサデバイスなどから取得された値が設定される．位置 r_i の現在の状態を，$v_c^{r_i} = \langle r_i.temp, r_i.humidity\rangle$ のように表し，k 個の位置に分割されたスマートスペース内の現在の環境を，$\langle v_c^{r_1}, \ldots, v_c^{r_k}\rangle$ と表す．

以上のことをふまえて，m 個のデバイス，l 人のユーザ，k カ所の位置の状態を保持するスマートスペースの状態（コンテキスト）c を，$c = \langle v_c^{d_1}, \ldots, v_c^{d_m}, v_c^{u_1}, \ldots, v_c^{u_l}, v_c^{r_1}, \ldots, v_c^{r_k}\rangle$ のように表す．

3.1.2 コンテキスト遷移

スマートスペースのコンテキストの遷移は，ノードの集合 V とエッジ（有向辺）の集合 E からなる有向グラフ $G = (V, E)$ で表すことができる．ここで，ノードはスマートスペースのコンテキスト，有向辺は 2 つのコンテキスト間の遷移に対応している．また，各有向辺には，2 つのコンテキスト間の遷移を可能にするイベントとコストが関連付けられる．しかしながら，デバイスの属性，ユーザの位置，環境の属性を連続値として扱うとコンテキストとして扱う範囲が無限になる．また，使用するデバイスの仕様や

© 2017 Information Processing Society of Japan

性能（センシング可能な値の範囲および分解能）に合わせて上限と下限を持った離散値として扱うことが可能であるが，設定可能な，または取得可能な値をすべてコンテキストとして扱うと，同様にコンテキスト空間は膨大になり，コンテキスト空間を探索するためのシステムの負担が大きくなる．この問題に対し，本論文では，コンテキストの各属性を，有限個の連続的な範囲（レンジ）に適切に分割することで対処する．たとえば，温度のレンジは，$(-\infty, 0]$，$(0, 10]$，$(10, 20]$，$(20, 30]$，$(30, \infty)$ のような有限個のレンジに区切ることができる．同様にユーザの位置に関してもスマートスペースの空間全体を適切に分割することで，有限個の部分空間の集合 R として扱う．

本研究では，**非環境イベント**，**環境デバイスイベント**，**環境遷移イベント**という 3 種類のイベントを考える．非環境イベントは，テレビのようなスマートスペース内の環境に影響を与えないデバイスの属性に値を設定するイベントである．環境デバイスイベントは，エアコンや加湿器といったスマートスペース内の環境に影響を与えるデバイスの属性を設定するイベントである．一方，環境遷移イベントは，温度や湿度といった環境に関する属性が，あるレンジから異なるレンジに遷移することによるコンテキストの遷移を表すイベントである．

デバイスの属性を直接設定できる非環境/環境デバイスイベントを，デバイス d_i の属性 j と代入を表す $\leftarrow$（左矢印），そして属性に設定する値 $VALUE$ を用いて，$E\{A^{d_i}(j) \leftarrow VALUE\}$ と表す．また，値を直接設定できない環境遷移イベントを，遷移前のコンテキスト c と遷移先のコンテキスト c'，そして遷移を表す $\rightarrow$（右矢印）を用いて，$E\{c \rightarrow c'\}$ と表す．

3.2 問題定義

ここでは，あるコンテキストから任意の別のコンテキストに最小コストで遷移できるイベント列を導出する問題を定義する．

イベント e によるコンテキスト c から c' への遷移を $c \xrightarrow{e} c'$ と表す．$path(c_s, c_t) = \langle e_1, .., e_n \rangle$ は，ソースコンテキスト c_s から対象コンテキスト c_t にスマートスペースを遷移させるイベント列を表す．$c_1, \ldots, c_{n+1}$ は，c_s と c_t の間で経由する中間コンテキストを表す．ここで，$c_1 = c_s$，$c_{n+1} = c_t$ として扱う．c_s から c_t への遷移可能なイベント列の条件を以下に記す．

$$\begin{aligned} &\exists c_2, \ldots, \exists c_n, \exists e_1, \ldots, \exists e_n \\ &c_s \xrightarrow{e_1} c_2 \wedge c_2 \xrightarrow{e_2} c_3 \wedge \ldots \wedge c_n \xrightarrow{e_n} c_t \\ &\quad \wedge e_1 \in E(c_s) \wedge .. \wedge e_n \in E(c_n) \end{aligned} \tag{1}$$

ここで，$E(c)$ は，コンテキスト c で実行可能なイベントの集合を表す．

本論文では，コンテキスト遷移に必要な消費電力量の最小化を目的にする．そのため，$e.cost$ をイベント e の消費電力量とすると，目的関数は以下のように表される．

$$\textbf{minimize} \sum_{i=1}^{n} e_i.cost \ \text{ subject to (1)} \tag{2}$$

各イベントの消費電力量は，デバイスごとにあらかじめ与えられた消費電力モデルを利用して計算するものとする．この消費電力モデルの構築方法として，実際にモニタリングした値を用いて構築する方法やカタログスペックを基にして構築する方法などが考えらえる．

4. PathSim：最小コストデバイス操作系列導出ツール

本章では，提案手法である最小コストデバイス操作系列（イベント系列）導出ツール PathSim について詳細を述べる．

PathSim は，任意の 2 つコンテスト間を最小コストで遷移させるデバイス操作系列を導出するツールである．本ツールは，コンテキストアウェアシステムの一部やコンテキストアウェアシステムをテストするツールなどに利用することを想定している．たとえば，コンテキストアウェアシステムにおいては，ユーザが入室した時点やユーザが指定した周期，また，窓やドアの開閉状況やデバイスの位置が変化した時点などで，現在のコンテキストから対象のコンテキストへ遷移するために利用できる．また，コンテキストアウェアシステムのテストにおいては，1 つのテストが終了した時点で，次のテスト環境までスマートスペースのコンテキストを遷移するために利用できる．

PathSim は任意に与えられたソースコンテキスト c_s と対象コンテキスト c_t 間の最小コストのデバイス操作系列を導出するために A*アルゴリズムを利用する．A*アルゴリズムは，ソースコンテキストから隣接するコンテキストを探索することで，経路を導出する手法であるが，対象とする問題では，遷移に必要なイベントおよびコストがあらかじめ与えられないため，隣接するコンテキストに遷移可能かどうか分からない．そのため，A*アルゴリズムを直接適用できないという問題がある．そこで PathSim では，探索を行う各コンテキストにおいて，シミュレータを用いてイベントを模擬実行することで，遷移する隣接コンテキストと遷移コストを導出し，A*アルゴリズムをベースに最小コストのデバイス操作系列を導出する．

PathSim では，3 章で定義したように，非環境イベント，環境デバイスイベント，環境遷移イベントという 3 つのイベントを用いてコンテキストを遷移させる．TV の操作のような非環境イベントはすぐに反映できるコンテキスト属性を対象とするため，操作系列の順番を考慮する必要はない．一方，温度や湿度を変化させる環境デバイスイベントにかかる消費電力量は，使用する機器，タイミング，稼働

© 2017 Information Processing Society of Japan

時間に依存する．そのため，イベント選択や実行順序を適切に行うことで，対象コンテキストに到達するために必要なコストを最小化する．

PathSim は，まず非環境イベントに関する操作系列 $non_env_path(c_s, c_r)$（以下，非環境イベント列と表す）を導出する．ここでコンテキスト c_r は，すべての非環境イベントを反映したあとのコンテキストを表す．その後，環境イベントのみによって構成される c_r から対象コンテキスト c_t へのイベント列 $env_path(c_r, c_t)$（以下，環境イベント列と表す）を導出する．PathSim は，それら 2 つのイベント列を統合することで，c_s から c_t に最小コストで遷移可能なパス $path_{(c_s,c_t)}$ を導出する．以下の節で，非環境イベント列と環境イベント列の導出手法の詳細を述べる．

4.1 非環境イベント列の導出法

PathSim は，c_s と c_t の非環境デバイスの各属性を比較し，その差に基づいて，非環境イベント列 $non_env_path(c_s, c_r)$ を導出する．PathSim は，最初に各非環境デバイス $d_i \in D$ について，属性値の組 $v^{d_i}_{c_s}$ と $v^{d_i}_{c_t}$ を比較する．j 番目の属性に差がある場合，すなわち，$v^{d_i}_{c_s}(j) \neq v^{d_i}_{c_t}(j)$ である場合には，イベント $E\{A^{d_i}(j) \leftarrow v^{d_i}_{c_t}(j)\}$ をイベント列に追加する．たとえば，c_s のテレビの状態が $v^{TV}_{c_s}(Status) = ON$ であり，c_t のテレビの状態が $v^{TV}_{c_t}(Status) = OFF$ である場合には，非環境イベント $E\{A^{TV}(Status) \leftarrow OFF\}$ がイベント列に追加される．また，イベント列の導出においては，テレビのチャネルを設定する前に，まずテレビの電源を “ON” に設定する，というようなデバイスの操作順序を考慮する．

4.2 環境イベント列の導出法

c_r から c_t への環境イベント列 $env_path(c_r, c_t)$ を導出する．

スマートスペースのコンテキストの遷移（コンテキスト，イベント，そしてイベントに対応したコスト）がすべて既知である場合には，コンテキストをノード，有向辺をイベントとコストとした有向グラフで表すことができる．グラフの 2 つのノード間の最小コストパスを見つける問題は最短経路問題と呼ばれ，その効率良い解法として A*アルゴリズム [10] が知られている．

PathSim では，環境イベント列 $env_path(c_r, c_t)$ を導出するために，A*アルゴリズムで使用する有向グラフにおける各辺の存在があらかじめ分かっていないということを考慮して A*アルゴリズムを改変する．また，コスト関数として本研究に適したものを採用する．以上のことをふまえ，PathSim は，スマートスペース内に設置されているデバイスに関する情報と，与えられた 2 つのコンテキストを用いて，スマートスペースのコンテキスト遷移（コンテキスト，イベント，そしてイベントに対応したコスト）をオンザフライに導出する．さらに，PathSim は，スマートスペースが対象コンテキストに到達するまで，環境デバイスイベントと環境遷移イベントを適切に選択する．

提案するパス導出アルゴリズムは，A*アルゴリズムの探索，コスト計算，パスの導出に関連する（a）PathFinder，（b）イベントジェネレータ，（c）コンテキスト遷移シミュレータ，（d）ActCost 関数，（e）EstCost 関数，という 5 つの関数で構成される．PathFinder は，A*アルゴリズムの基本的な処理を行うメイン関数であり，残りの 4 つの関数を用いて，パス，すなわち，環境イベント列 $env_path(c_r, c_t)$ を導出する．イベントジェネレータは，任意のノード v_c から隣接ノードを探索するために，実行可能な環境イベントの集合 $E(c)$ を生成する．コンテキスト遷移シミュレータは，イベント $e \in E(c)$ を，コンテキスト c において実行したときのコンテキストの遷移をシミュレートすることで隣接ノードを導出する．ActCost 関数と EstCost 関数は，任意のノード v_c のコストを計算するために用いられる．ActCost 関数は，c_r から c までコンテキストを遷移させるための，実コストを計算する関数を表し，EstCost 関数は，c から c_t までの推定コストを計算する関数を表す．任意のノード v_c のコストは，実コストと推定コストの合計コストである．

以降では，4.2.1〜4.2.3 項において，イベントジェネレータ，コンテキスト遷移シミュレータ，各コスト関数について詳細を述べ，これらの関数を利用し PathFinder が環境イベント列を導出するプロセスを 4.2.4 項において確認する．

4.2.1 イベントジェネレータ

この関数は，任意のコンテキスト c において実行可能なイベントの集合を生成する関数である．スマートスペース内には，複数のデバイスが設置されていることから，単に c において実行可能なイベントをすべて生成する方法では，解空間と計算時間が肥大化する．そのため，対象コンテキスト c_t に到達させるために，c において**有効な**（または，利用可能な）イベントのみを生成するようにする．本論文では，加湿器（HM），除湿機（DH），ヒータ（HT）の電源を ON にするイベントを，それぞれ，加湿，除湿，温度の上昇に**有効なイベント**と定義する．また，エアコン（AC）に関しては，冷房，暖房，ドライ（除湿）といった複数のモードを持つと考えられるため，状態の属性 $A^{AC}(Status)$ の値域を $Dom(A^{AC}(Status)) = \{OFF, Cold, Hot, Dehumid\}$ と定義し，モードを切り替えるイベントを，それぞれ，温度の低下，温度の上昇，湿度の低下に有効なイベントと定義する．たとえば，コンテキスト $c = ([22°\mathrm{C}, 23°\mathrm{C}), [40\%, 45\%))$ から，$c_t = ([25°\mathrm{C}, 26°\mathrm{C}), [50\%, 55\%))$ に向かってコンテキストを遷移させる場合には，温度や湿度を c_t に向かって変化させることが可能なヒータの電源を入れるイベント $E\{v^{HT}(Status) \leftarrow \text{"}ON\text{"}\}$，エアコンを暖房にするイベント $E\{v^{AC}(Status) \leftarrow \text{"}Hot\text{"}\}$，そして，加湿器の電源を入

© 2017 Information Processing Society of Japan

れるイベント $E\{v^{HM}(Status) \leftarrow \text{"}ON\text{"}\}$ といったイベントが生成される．一方，除湿機の電源を ON にするイベントやエアコンを冷房モードにするイベントなどの，c_t から遠ざかるようにコンテキストが遷移する可能性があるイベントは有効なイベントとして定義されていないため，生成されることはなく，有効ではないイベントが実行済みとなっている場合には，電源を OFF にするイベントが生成される．

3.1.2 項で述べたように，温度/湿度の変化によるコンテキストの遷移は，環境デバイスイベント（デバイスの操作）では行うことができないため，遷移前と遷移後のコンテキストを用いて，環境遷移イベント $E\{c \rightarrow c'\}$ で表す．しかしながら，シミュレータによってイベントが模擬実行されるまでは，遷移後のコンテキストは分からない．そこで，イベントジェネレータは，直接的にイベントおよび遷移先のコンテキストを表すことのできない暗示的なイベント（Implicit event）を意味する $E\{\Im\}$ というイベントを生成する．

生成された各イベントは，次項で述べるシミュレータによって，コンテキスト c において，それぞれ，実行した場合に遷移するコンテキスト c' を導出するために用いられる．

4.2.2 コンテキスト遷移シミュレータ

コンテキスト遷移シミュレータは，$EventSimulator(c, e)$ という関数で実装される．この関数は，現在のコンテキスト c とイベント e を入力し，c において e を実行することで到達する隣接コンテキスト c' を出力する．イベント e が環境デバイスイベントである場合には，コンテキスト c' は e に関連した属性以外は c と同じ属性値を持つ．一方，e が環境遷移イベントである場合には，関数はスマートスペース内の温度と湿度の変化をシミュレーションし，c の温度か湿度が隣接するレンジに遷移した後のコンテキスト c' を導出する．

シミュレーションでは，スマートスペースの容積（サイズ），窓や天井を含めた壁の熱伝導率，外気温，デバイスが発する熱や水蒸気，などが考慮される．

温度変化をシミュレーションするために，シミュレータは空間内の温度 tmp が隣接する温度レンジの中央値 tmp_{nei} に到達するまでの時間 $T_{tmp \rightarrow tmp_{nei}}$ を，以下の式を用いて計算する．

$$T_{tmp \Rightarrow tmp_{nei}} = \frac{Q_{tmp \rightarrow tmp_{nei}}}{Q_{dev} + Q_{ext}} \tag{3}$$

where

$$Q_{tmp \rightarrow tmp_{nei}} = C \cdot (tmp_{nei} - tmp)$$
$$Q_{ext} = H \cdot Area \cdot (tmp_{ext} - tmp)$$

ここで，$Q_{tmp \rightarrow tmp_{nei}}$ は，空間内の温度を tmp から tmp_{nei} へ変化させるために必要な熱量を表し，温度差と空間の熱容量 C を乗算することで計算される．Q_{dev} は，温度に関連したすべての動作中のデバイスから時間あたりに得られる熱量を表している．Q_{ext} は外気温 tmp_{ext} から壁を通して時間あたりに得られる熱量を表しており，熱伝導率，すべての壁の面積 $Area$，そしてスマートスペース内の温度と外気温との差 $tmp_{ext} - tmp$ によって計算される．

湿度の変化をシミュレーションするために，空間内の現在の湿度 hmd が隣接する湿度レンジの中央値 hmd_{nei} に到達するまでの時間 $T_{hmd \rightarrow hmd_{nei}}$ を計算する．この計算では，空間の体積 Vol，湿度に関するすべての動作中のデバイスが時間あたりに発する（または，除去する）水蒸気量 M_{dev} および 1 秒ごとに流入出する水蒸気量 M_{ext} を考慮し，以下の式を使用して時間を計算する．

$$T_{hmd \rightarrow hmd_{nei}} = \frac{Vol \cdot (hmd_{nei} - hmd)}{M_{dev} + M_{ext}} \tag{4}$$

where

$$M_{ext} = Vol_{ven} \cdot (hmd_{ext} - hmd)$$

ここで，M_{ext} は，1 秒あたりの換気量 Vol_{ven} および外気の湿度 hmd_{ext} と空間内の湿度 hmd の差によって計算される．この際，相対湿度は空間内の水蒸気量が同じであっても，温度が異なることで変化するため，上記の式では，Tetens の式 [12] を利用して，相対湿度を絶対湿度に変換して計算を行う．

$T_{tmp \rightarrow tmp_{nei}}$ と $T_{hmd \rightarrow hmd_{nei}}$ を計算した後，シミュレータは両方の値を比較する．このとき，$T_{tmp \rightarrow tmp_{nei}} < T_{hmd \rightarrow hmd_{nei}}$ である場合には，温度が湿度より先に隣接するレンジに到達するため，シミュレータは以下の式を用いて現在の湿度 hmd の値を更新する．

$$hmd \leftarrow hmd + \frac{((M_{dev} + M_{ext}) \times T_{tmp \rightarrow tmp_{nei}})}{Vol}$$

一方，$T_{tmp \rightarrow tmp_{nei}} > T_{hmd \rightarrow hmd_{nei}}$ である場合には，シミュレータは以下の式を使用して現在の温度 tmp を更新する．

$$tmp \leftarrow tmp + \frac{((Q_{dev} + Q_{ext}) \times T_{hmd \rightarrow hmd_{nei}})}{C}$$

更新された tmp と hmd の値は，到達したコンテキスト c' に，$c'.tmp$ と $c'.hmd$ としてそれぞれ保存される．

その後，シミュレータは，以下の式を用いて，遷移時間 T において動作中の全デバイスの消費電力量の総和を求め，コストとして $e.cost$ に保持する．

$$e.cost = \sum_{d \in D} \int_0^T (power(d) \cdot t) dt \tag{5}$$

ここで T は，$T_{tmp \rightarrow tmp_{nei}}$ と $T_{hmd \rightarrow hmd_{nei}}$ のうち小さい方の時間であり，$power(d)$ は，デバイス d の消費電力を表す．デバイス d を動作させたときの消費電力 $power(d)$ は，デバイスごとにあらかじめ与えられた消費電力のモデルを用いて計算する．

© 2017 Information Processing Society of Japan

4.2.3 コスト関数

4.2.4 項で説明する PathFinder では，各列のコストを評価するために，$ActCost(c, c')$ と $EstCost(c', c_t)$ という 2 つのコスト関数を使用する．$ActCost(c, c')$ はコンテキスト c から c' に遷移する際の，イベントの合計コストを計算する関数である．前項で説明したように，コンテキスト遷移シミュレータを用いて，コンテキストの遷移にかかるコストを計算できる．そのため，c から c' までのパスに含まれるすべてのイベントのコストを足し合わせることでパスのコストを計算できる．一方，$EstCost(c', c_t)$ は，探索が行われていないコンテキスト c' から対象コンテキスト c_t へのコストを推定する関数である．$EstCost()$ は，c' から c_t へ遷移する間の操作系列を考慮せず，c' と c_t 間の遷移に必要な熱量および水蒸気量に対して，c' から有効なデバイスを動作させた場合のデバイスが発する熱量と水蒸気量を用いて，それぞれの遷移に必要な消費電力量を計算し合計するでコストの推定を行う．この際，温度および湿度にそれぞれ有効なデバイスに対して，使用する場合と使用しない場合の組合せの中から，遷移に必要な消費電力量が，最小の消費電力量を推定コストとして採用する．

4.2.4 PathFinder：環境イベント列導出アルゴリズム

PathSim のメイン関数である PathFinder は，コンテキスト c_r から c_t への最小コストパス（イベント列）を A*アルゴリズムに基づいて導出する．

PathFinder によって導出される各ノードは，それぞれ，コンテキスト $v.c$，親ノード $v.p$，$v.p$ から $v.c$ に遷移するために実行されたイベント $v.e$，$v.p$ から $v.c$ に遷移するためにかかるコスト $v.cost$，そして v_r から v までのコストと v から v_t への推定コストから計算される合計コスト $v.cost_{total}$ を保持する．v_{cur} を現在のノード，v_{nei} を v_{cur} からイベント $v_{nei}.e$ によって到達するノード，そして，v_{old} を v_{nei} より先に導出された同じコンテキストを持つノードと定義する．また，v_r と v_t を，それぞれソースコンテキスト c_r を持つソースノードと対象コンテキストを持つ c_t を持つ対象ノードとして定義する．

PathFinder のアルゴリズムを Algorithm 1 に示す．PathFinder のアルゴリズムは，c_r と c_t が入力されたときに，コンテキスト c_r から c_t に最小コストで遷移可能なイベント列 $env_path(c_r, c_t)$ を導出し，出力するアルゴリズムである．

以下では，アルゴリズムの核となる部分に関して簡潔に説明する．PathFinder は，まず，現在のノードが保持するコンテキスト $v_{cur}.c$ において実行可能な環境イベントの集合 $E(v_{cur}.c)$（以下，$E()$）を生成する（line 8）．集合 $E()$ には，$\{E\{\Im\}\}$ で表される環境遷移イベントも含まれている．このイベントは，現在のコンテキスト $v_{cur}.c$ から，温度と（または）湿度のレンジが異なる隣接コンテキスト $v_{nei}.c$ にスマートスペースが遷移することを表す．各環境

Algorithm 1 PathFinder

Input: c_r, c_t
Output: $env_path(c_r, c_t)$

1: $v_s.c \leftarrow c_r$, $v_s.p \leftarrow null$, $v_s.e \leftarrow null$, $v_s.cost \leftarrow 0$, $v_s.cost_{total} \leftarrow 0$
2: $v_t.c \leftarrow c_t$, $v_t.p \leftarrow null$, $v_t.e \leftarrow null$, $v_t.cost \leftarrow 0$, $v_t.cost_{total} \leftarrow 0$
3: $OPEN \leftarrow \{v_s\}$, $CLOSE \leftarrow \emptyset$
4: $v_{cur} \leftarrow \mathrm{argmin}_{v \in OPEN} v.cost_{total}$, $v_{nei} \leftarrow null$, $v_{old} \leftarrow null$
5: **while** $v_{cur}.c \neq v_t.c$ **do**
6: $\quad CLOSE \leftarrow CLOSE \cup \{v_{cur}\}$
7: $\quad OPEN \leftarrow OPEN \setminus \{v_{cur}\}$
8: $\quad E() \leftarrow EventGenerator(v_{cur}.c, v_t.c)$
9: $\quad$ **for all** $e \in E()$ **do**
10: $\quad\quad v_{nei} \leftarrow EventSimulator(v_{cur}, e)$
11: $\quad\quad v_{nei}.cost_{total} \leftarrow ActCost(v_s, v_{nei}) + EstCost(v_{nei}, v_t)$
12: $\quad\quad$ **if** $\neg\exists v \in OPEN \cup CLOSE\ (v.c = v_{nei}.c)$ **then**
13: $\quad\quad\quad OPEN \leftarrow OPEN \cup \{v_{nei}\}$
14: $\quad\quad$ **else if** $\exists v \in OPEN\ (v.c = v_{nei}.c)$ **then**
15: $\quad\quad\quad v_{old} \leftarrow v$
16: $\quad\quad\quad$ **if** $v_{nei}.cost_{total} < v_{old}.cost_{total}$ **then**
17: $\quad\quad\quad\quad OPEN \leftarrow OPEN \setminus \{v_{old}\}$
18: $\quad\quad\quad\quad OPEN \leftarrow OPEN \cup \{v_{nei}\}$
19: $\quad\quad\quad$ **end if**
20: $\quad\quad$ **else if** $\exists v \in CLOSE\ (v.c = v_{nei}.c)$ **then**
21: $\quad\quad\quad v_{old} \leftarrow v$
22: $\quad\quad\quad$ **if** $v_{nei}.cost_{total} < v_{old}.cost_{total}$ **then**
23: $\quad\quad\quad\quad CLOSE \leftarrow CLOSE \setminus \{v_{old}\}$
24: $\quad\quad\quad\quad OPEN \leftarrow OPEN \cup \{v_{nei}\}$
25: $\quad\quad\quad$ **end if**
26: $\quad\quad$ **end if**
27: $\quad$ **end for**
28: $\quad$ **if** $OPEN = \emptyset$ **then**
29: $\quad\quad$ **search failed**
30: $\quad$ **else**
31: $\quad\quad v_{cur} \leftarrow \mathrm{argmin}_{v \in OPEN} v.cost_{total}$
32: $\quad$ **end if**
33: **end while**
34: $env_path(c_r, c_t) \leftarrow \epsilon$, where ϵ is the empty string
35: $env_path(c_r, c_t).cost \leftarrow v_{cur}.cost_{total}$
36: **while** $v_{cur} \neq v_s$ **do**
37: $\quad$ {Traversing nodes from v_{cur} *to* v_s to derive the path $env_path(c_r, c_t)$}
38: $\quad env_path(c_r, c_t) \leftarrow \langle v_{cur}.e \rangle.env_path(c_r, c_t)$
39: $\quad v_{cur} \leftarrow v_{cur}.p$
40: **end while**
41: **return** $env_path(c_r, c_t)$

遷移イベントに対して，PathFinder はイベントを実行した後の隣接コンテキスト v_{nei} を導出する（line 10）．その後，$v_s.c$ から $v_{nei}.c$ に到達するためのコストをコスト関数 $ActCost(v_s.c, v_{nei}.c)$ を用いて計算し，$v_{nei}.c$ から $v_t.c$ に到達するための推定コストをコスト関数 $EstCost(v_{nei}.c, v_t.c)$ を用いて計算する（line 11）．PathFinder は，計算された 2 つのコストを合計することで，$v_s.c$ から $v_t.c$ に $v_{nei}.c$ を経由して到達するための合計コストを決定する．PathFinder は，最小合計コストを持つノードを v_{cur} として再帰的に選択し，選択された v_{cur} ごとに隣接ノードを導出する．この

© 2017 Information Processing Society of Japan

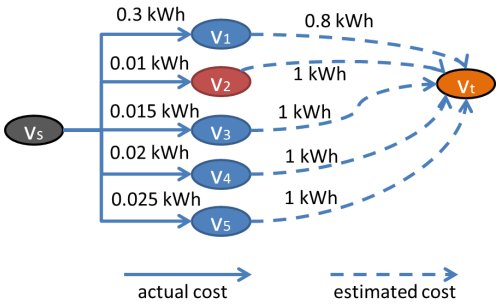

図 1　v_{nei} の導出と v_{cur} の選択の例

Fig. 1　Neighboring nodes are calculated with the total costs to reach the target node.

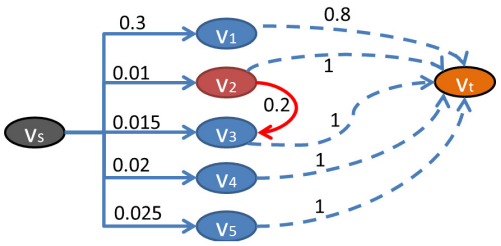

図 2　v_{nei} がすでに *OPEN* または *CLOSE* に含まれているケース

Fig. 2　The case when a neighboring node is included in OPEN or CLOSE.

処理を v_t が v_{cur} として選択されるまで行う（lines 12–32）．

アルゴリズムの動作を，**図 1** と**図 2** の例を用いて説明する．図 1 では，未探索ノードの集合 *OPEN* から，探索を行うノードとして v_s が選択され，イベントジェネレータにより 5 つのイベントを含む $E()$ が生成される．その後，PathFinder は，$E()$ の各イベントに対して，$EventSimulator()$ 関数を用いて，隣接コンテキストノード v_{nei} を決定し（line 10），v_{nei} を経由した場合の v_s から v_t へ遷移するために必要な合計コストを計算する（line 11）．図 1 では，v_s から到達可能な v_1 から v_5 までの 5 つの隣接ノードが合計コストとともに導出される．この例では，v_1 から v_5 までの各ノードに関して，v_s から各ノードに到達するパスが 1 つしかないため，line 12 で表されている条件と合致する．そのため，v_1 から v_5 までの 5 つの隣接ノードは *OPEN* リストに追加される（line 13）．

2 回目のループ（lines 5–33）では，*OPEN* 内で 1.01 kWh という最も低いコストを持つ v_2 が v_{cur} として選択される（line 4）．そして，$E()$ の生成，隣接ノードの導出などの上記の処理が繰り返される．

しかし，隣接ノード v_{nei} がすでに未探索ノードの集合 *OPEN* か探索済みノードの集合 *CLOSE* に含まれている場合（lines 14，20），v_{nei} には以前の探索によって異なるパスと合計コストがすでに保持されているといえる．簡単に説明するため，以前の探索で導出されたノードを v_{old} と表す．PathFinder は，v_{nei} を通る現在のパスと v_{old} を通る以前のパスのうち，最小合計コストを持つノードのみを保持する．このとき，v_{nei} を保持する場合には，v_{old} を *OPEN* か *CLOSE* から除外し，v_{nei} を *OPEN* に追加する（lines 16–19，22–25）．

図 2 は v_{nei} が *OPEN* にすでに含まれている場合を表している．このケースで v_3 は，v_2 からコスト 0.2 kWh で到達可能であるが，v_3 はすでに *OPEN* に含まれている．ここで，v_2 から到達する v_3 を v_{nei} で表し，v_s から到達する v_3 を v_{old} と表す（lines 14，15）．PathFinder は，2 つのノードの合計コストを比較し，最も小さい合計コストを持つ方を *OPEN* に保持する（lines 16–19）．図 2 の場合，$v_s \rightarrow v_3(v_{old}) \rightarrow \ldots \rightarrow v_t$ の合計コストは $0.015 + 1 = 1.015$ kWh，$v_s \rightarrow v_2 \rightarrow v_3(v_{nei}) \rightarrow \ldots \rightarrow v_t$ の合計コストは $0.01 + 0.2 + 1 = 1.21$ kWh である．そのため，v_{old} が *OPEN* に保持され続け，v_{nei} は *OPEN* に追加されず破棄される．Algorithm 1 の lines 20–26 で表されているように，$v_{nei}.c$ がすでに *CLOSE* に含まれている場合にも同様の処理が適用される．ただし，$v_{nei}.c$ を保持することが決定された場合には，*CLOSE* に含まれている v_{old} を破棄し，$v_{nei}.c$ を *OPEN* に追加する．

上記の探索は，対象ノードに到達するまで繰り返される．すなわち，v_{cur} と v_t が同じ場合に，アルゴリズムは lines 5–33 のループを終了する．イベント列 $env_path(c_r, c_t)$ は，対象ノード v_t（$= v_{nei}$）から v_s に到達するまで親ノード $v_{cur}.p$ を再帰的にたどりながら，各ノードのイベント $v_{cur}.e$ をパスの先頭に追加していくことで導出される．

探索するノードが *OPEN* リストに存在しない場合には，対象ノードへのパスが存在しないと判定し，探索を終了する．この場合には，イベント列は出力されず，探索が失敗したことが通知される．

4.3　非環境/環境イベント列の結合

$non_env_path(c_s, c_r)$ と $env_path(c_r, c_t)$ という 2 つのイベント列を導出した後に，それらを結合することでデバイス操作系列 $path(c_s, c_t)$ を獲得できる．このとき，非環境イベントは，ユーザがすぐに提供されることを望むサービスだと考えられる．たとえば，テレビの電源を “ON” にするイベントは，スマートスペース内の環境を変化させる前に実行されるべきイベントである．したがって，$env_path(c_r, c_t)$ は $non_env_path(c_s, c_r)$ の後に実行されるように結合される．

5.　シミュレーションによる評価

本章では，日本の夏と冬におけるスマートホームを想定したシミュレーションによって，PathSim で導出したデバイス操作系列の消費電力量を評価する．

5.1　実験設定

以下に，本実験のシミュレーションで使用するスマート

© 2017 Information Processing Society of Japan

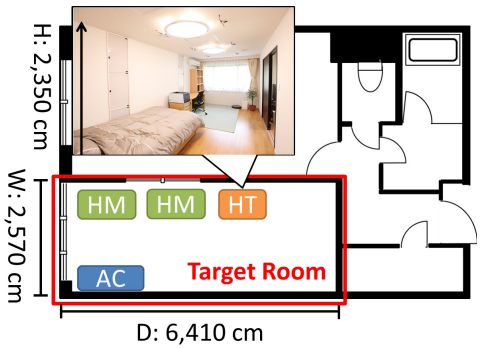

図 3　対象とするスマートホーム
Fig. 3　Configuration of the target room.

表 1　実験で使用するデバイスの設定
Table 1　Settings of the appliances used during the experiment.

	消費電力	W：冷/暖房能力，L：加/除湿量
エアコン	COLD：990 W, HOT：1,225 W, DH：800 W	COLD：3.4 kW, HOT：4.7 kW, DH：3 L/h
ヒータ	1,500 W	1,500 W
除湿器	600 W	1,440 mL/h
加湿器-1	100 W	1,260 mL/h
加湿器-2	90 W	1,080 mL/h

ホームのデバイスと遷移させるコンテキストの設定，そして実験方法を記す．

5.1.1　対象とするスマートホームおよび使用するデバイス

対象とするスマートホームとして，**図 3** で表したスマートホームのフロアプランのうち，簡単化のため，図内の赤色の線で囲まれた 1 つの部屋のみを対象とする．対象空間を囲む各壁の熱伝導率として，木製の壁と天井の熱伝導率を $0.15W/(m \cdot K)$，ガラス窓の熱伝導率を $0.2W/(m \cdot K)$ とした．また，流入出する水蒸気量を計算するために，予備実験として図 3 の対象空間において湿度の変化量を実際に計測し，1 秒あたりの換気量 $Vol_{ven} = 0.0762$ を導出した．本実験では，窓やドアの開閉状況が変化しないコンテキスト間の遷移を想定し，上記の換気量も変化しないものとする．

本実験では，1 台のエアコン，1 台のヒータ，1 台の除湿器，2 台の加湿器という 5 つのデバイスを使用する．**表 1** は，各デバイスの消費電力と各種能力を表しており，これらに基づき，温度・湿度の遷移と遷移コストのシミュレーションを行う．さらに，コンテキストアウェアシステム自体の動作コストとして，毎秒 15 W の消費電力が消費されるものとする．これにより，デバイスが何も稼働していなくても，環境遷移イベントにより遷移させた場合には，時間が経過するごとにコストが加算される．

5.1.2　ソース/対象コンテキスト

本実験では，夏と冬においてユーザの好みの環境にスマートホームを調整するコンテキストアウェアシステムを想定し評価を行う．

対象コンテキストは，室内環境を専門とする温度計・湿度計を製造するメーカが公開している「快適な温度・湿度の目安」[13] を参考に生成する．夏の快適な温度 [25°C, 28°C) を 1°C ごとに区切ることで 3 つのレンジ $\langle[25°C, 26°C), [26°C, 27°C), [27°C, 28°C)\rangle$ に分割し，夏の快適な湿度 [55%, 65%) に関しても 2.5%ごとに区切ることで 4 つのレンジ $\langle[55\%, 57.25\%), [57.25\%, 60\%), [60\%, 62.25\%), [62.25\%, 65\%)\rangle$ に分割する．また，冬の快適な温度 [18°C, 21°C) と快適な湿度 [47.5%, 57.5%) も同様に 3 つの温度レンジと 4 つの湿度レンジに分割する．そして，温度と湿度のペアを用いて季節ごとに 12 個の対象コンテキストを生成する．

一方，ソースコンテキストは，冬に関しては，図 3 の対象スペースで実際にモニタリングされた室温を参考に [15°C, 18°C) を，夏に関しては，奈良県の気象情報を参考に [28°C, 31°C) をそれぞれ 3 つの温度レンジに分割し，湿度に関しては，モニタリングした値と気象情報を参考に，夏に関しては [60%, 70%) を，冬に関しては [40%, 50%) をそれぞれ 4 つの湿度レンジに分割し，対象コンテキストと同様に温度と湿度のペアを用いて季節ごとに 12 個のソースコンテキストを生成する．

本実験では，外気温と湿度は，ソースコンテキストと同じ値を使用し，時間によって変化しないものとする．また，ソースコンテキストと対象コンテキストにおいて，すべてのデバイスの電源状態は OFF に設定され，コンテキスト間でデバイスが新たに追加されたり，取り除かれたりするようなデバイスの位置情報の変化は考慮しない．

5.1.3　実験手法

本実験では，季節ごとに生成した 12 個のソースコンテキストと 12 個の対象コンテキストに対して，12 × 12 の 144 通りのデバイス操作系列を導出し，各操作系列の消費電力量や遷移時間を評価する．

本実験では，**図 4** に表したハイブリッドオートマトンに基づき各デバイスが独立して動作を行うハイブリッドダイナミカルシステム（Hybrid Dynamical System，以下，HDS と表記する）と，消費電力あたりのデバイス能力を考慮し，任意のコンテキストにおいて対象コンテキストに遷移するために最も効率の良いデバイスを選択して使用する手法（Selecting Most Efficient Device，以下，SMED と表記する）を比較手法とする．

HDS [14] は，温度や湿度などの連続値とデバイスの状態などの離散値を扱う動的システムであり，空調機器の制御で一般的に使用されている．本実験で HDS は，図 4 で表したハイブリッドオートマトンに基づき制御を行う．図 4 のハイブリットオートマトンにおいて，デバイスの状態はモード（コンテキスト），各エッジの添え字および各モードの下部に記述された条件式は遷移条件（遷移を許可する条

© 2017 Information Processing Society of Japan

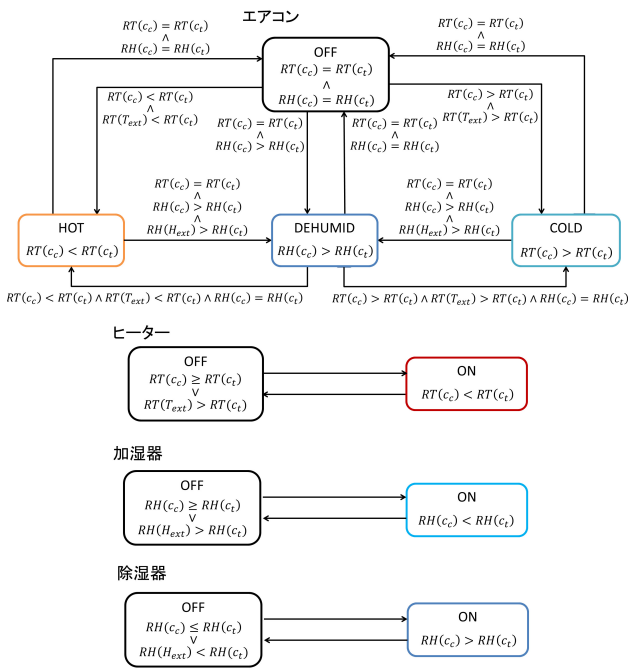

図 4　HDS で使用する各デバイスのオートマトン
Fig. 4　Automaton of each appliance.

件）と不変条件（モードを維持するために満たすべき条件）を表す．また，ハイブリッドオートマトンは，各モードにおいて，変数として温度，湿度，消費電力などの連続値，およびこれらの変化量を保持するが，図 4 では簡略化と省略を行い遷移に関する条件のみを表記している．図 4 における $RH(c)$，$RT(c)$ はコンテキスト c における湿度レンジと温度レンジを表し，c_c と c_t は現在のコンテキストと対象コンテキスト，T_{ext}，H_{ext} は外気温と外湿度を表している．また，各モードの不変条件は，モードを維持するために必要な条件が表されており，条件を満たさなくなった時点で必ず遷移が行われることを表している．たとえば，ヒータの OFF モードは，現在の温度レンジ $RT(c_c)$ が対象コンテキスト c_t の温度レンジと同じ場合か，より高い温度を表すレンジの場合，または対象コンテキストの温度レンジが表す温度より外気温の方が高い場合にのみ維持される．これは，夏場に温度を上げる場合に，一般的にヒータを使わず外気からの影響のみで上昇させるように，外気によって対象コンテキストに遷移可能な場合には OFF モードを維持し，現在の温度と外気温が低い場合にヒータが ON モードに遷移することを表している．エアコンに関しても，外気のみで遷移できる場合には OFF モードを維持するが，簡単のため図 4 では省略し遷移条件のみを記述した．

SMED では，任意のコンテキストにおいて，対象コンテキストに遷移するために必要な冷暖/暖房，加湿/除湿能力などのデバイス能力を各デバイスに対して評価し，評価対象ごとに最も効率の良いデバイスを 1 つ選択する．本実験では，温度と湿度に対して，効率の良いデバイスが 1 つずつ選択される．ただし，エアコンのような冷房，暖房，除湿機能を持ったデバイスについては，OFF モードの場合や評価対象の機能が有効になっている場合にのみ評価に含める．たとえば，除湿の能力を評価する際に，すでに冷房モードで稼働しているエアコンは，除湿効率が最も良かったとしても，評価の対象とせず，2 番目に除湿効率が良いデバイスを選択する．この際，温度が目的のコンテキストに到達し，エアコンが OFF モードに切り替わった場合には，再度評価を行い，2 番目に除湿効率が良いデバイスから最も除湿効率が良いエアコンへ除湿に使用するデバイスを切り替える．

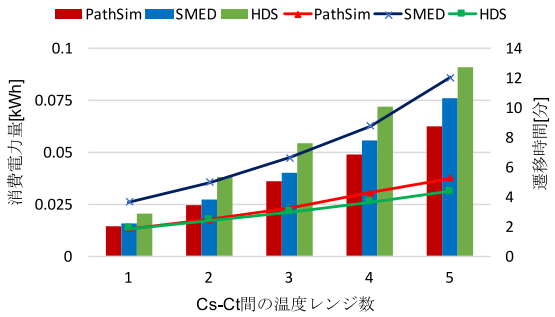

図 5　冬想定時のデバイス操作系列の消費電力量および遷移時間
Fig. 5　Energy consumptions and transition times during winter simulation.

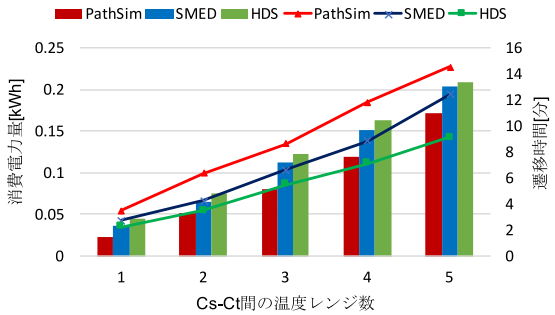

図 6　夏想定時のデバイス操作系列の消費電力量および遷移時間
Fig. 6　Energy consumptions and transition times during summer simulation.

5.2　実験結果

実験結果を**図 5**，**図 6** に示す．図 5 と図 6 の棒グラフはレンジ数ごとの消費電力量の平均を，折れ線グラフは遷移時間の平均を表している．PathSim で導出したデバイス操作系列は，HDS と SMED で導出したデバイス操作系列と比べて少ない消費電力量でコンテキストを遷移できることが確認できる．また，遷移時間に関しては，冬を想定した実験結果において，SMED より短い時間で遷移し，すべてのデバイスを使用する HDS と同様の遷移時間で，かつ少ない消費電力量で遷移できるデバイス操作系列を導出していることが分かる．一方，夏を想定した実験結果においても，SMED と HDS より少ない消費電力量でコンテキストを遷移できることが確認できるが，遷移時間に関しては，冬を想定した実験結果とは異なり，SMED と HDS より長

© 2017 Information Processing Society of Japan

表 2 比較手法に対する消費電力量の削減率
Table 2 Comparison of energy consumption for SMED and HDS.

		$c_s - c_t$ 間の温度レンジの数					
		1	2	3	4	5	平均
冬想定時	SMED	9.3%	8.5%	10.3%	12.7%	18%	11.8%
	HDS	31.7%	34.2%	33.5%	32.5%	31.5%	32.7%
夏想定時	SMED	40.1%	22.2%	28.1%	21.9%	16%	25.6%
	HDS	50.2%	31.9%	34.9%	27.1%	17.8%	32.4%

表 3 導出されたデバイス操作系列の平均操作数（() は環境デバイスイベントのみの平均操作数）
Table 3 Average number of events included in the event sequence, (): Average number of explicit events.

		$c_s - c_t$ 間の温度レンジの数					
		1	2	3	4	5	平均
冬想定時	PathSim	10.8 (5.8)	12.6 (6.4)	15.2 (7.4)	17.9 (8.4)	18.8 (7.9)	15 (7.2)
	SMED	9.3 (4)	11 (4)	12.6 (4.3)	15.6 (5.3)	19.8 (7)	13.6 (4.9)
	HDS	13 (8)	14.4 (8)	15.7 (8)	17 (8.1)	18.6 (8.3)	15.7 (8.1)
夏想定時	PathSim	10.9 (4.7)	14 (4.7)	16.5 (4.7)	21.5 (6.25)	26.1 (8.1)	17.8 (5.7)
	SMED	9.8 (5.5)	12 (5.9)	15 (6.8)	17 (7)	22.4 (9.1)	15.2 (6.9)
	HDS	9.8 (5.5)	11.9 (5.9)	15.2 (7)	16 (6.4)	18.3 (6.4)	14.2 (6.2)

くなっている．

また，**表 2** は比較手法に対する消費電力量の削減率を表している．冬を想定した実験結果では，SMED に対してはレンジ数が増加するほど削減率が高くなっており，平均約 12%の消費電力量を削減している．また，HDS に対しては一定して削減率が高く平均約 33%削減していることが分かる．一方で，夏を想定した実験結果では，SMED に対しては，冬を想定した実験結果より削減率が高く平均約 26%削減しており，HDS に対しても冬と同程度の削減率で平均約 32%削減している．しかしながら，冬想定時とは反対に，レンジ数が増加するほどに削減率の幅が少なくなっている．

表 3 は，各手法によって導出されたデバイス操作系列の平均操作数を表し，カッコ内の数字は環境デバイスイベントのみの平均操作数を表している．冬を想定した実験において，環境デバイスイベントの平均操作数が 4 である SMED の場合には，2 つのデバイス（エアコン，加湿器-1）の ON/OFF のみで目標のコンテキストに到達している．また，環境デバイスイベントの平均操作数が 8 である HDS の場合も，4 つのデバイス（エアコン，ヒータ，加湿器-1，2）の ON/OFF のみで目標のコンテキストに到達している．たとえば，SMED の場合，エアコンと加湿器-1 をはじめに起動し，目標の温度のレンジに達した時点でエアコンの電源を切り，続いて目標の湿度のレンジに達した時点で加湿器-1 の電源を切るといった操作が行われている．一方，PathSim においては，ヒータを除いた 3 つのデバイス（エアコン，加湿器-1，2）を操作し消費電力を削減している．また，レンジの数が 4 以上の場合には，PathSim はすべてのデバイスを使用する HDS と同程度の平均操作数になっているが，エアコンや加湿器 1，2 の ON/OFF を途中で切り替えるなど，細かな操作を行うことで消費電力を削減している．夏を想定した実験においては，SMED や HDS より環境デバイスイベントの操作数が少ないにもかかわらず，環境遷移イベントを含めた平均操作数が他の手法より多くなっている．夏を想定した実験では，温度と湿度を下げる能力を持つエアコンと除湿器の 2 つのデバイスしか使用されないため，レンジ数が少ない場合は，SMED と HDS の平均操作数はほとんど変わらない．SMED においてレンジ数が多くなった場合に平均操作数が多くなっているのは，エアコンの冷房モードで目標の温度レンジに遷移したのち，除湿で使用するデバイスを除湿器からエアコンに切り替えているためである．一方，PathSim では，レンジ数が少ない場合にはエアコンのみを使用しているため他の手法より環境デバイスイベントの平均操作数が少ない．しかし，エアコンのみで遷移させる場合には，レンジ数が多くなると，冷房または除湿中に一方のレンジが外気により元に戻る可能性が高くなる．そのため，PathSim では目標の湿度に達した後にも除湿を続けるような操作を行っている．これにより，他の手法と比べ環境遷移イベントを含めた平均操作数が多くなることで，図 6 の遷移時間が他の手法より長くなっていると考えられる．

上記のように，提案手法は省電力でコンテキストを遷移できる手法であるが，図 6 のように，遷移時間が多くかかる可能性も存在する．この問題は，使用できるデバイスを増やすことや，提案手法に時間制約を追加し，時間制約を満たすデバイス操作系列を導出することにより解決できる

© 2017 Information Processing Society of Japan

表 4　実験で使用するデバイスの設定
Table 4　Settings of the appliances used during the experiment.

	消費電力	W：冷/暖房能力，L：加/除湿量
エアコン-1	COLD：990 W，DH：800 W	COLD：3.4 kW，DH：3.0 L/h
エアコン-2		
エアコン-3		
除湿器	600 W	1,440 mL/h

表 5　選択可能なデバイス数と使用したデバイス
Table 5　Number of applliances and their types used for the expereiment.

デバイス数	使用デバイス
2	エアコン-1，除湿機
3	エアコン-1，2，除湿機
4	エアコン-1，2，3，除湿機

と考えられる．

5.3　デバイス数に応じた消費電力量の評価

夏想定時のデバイス数に応じた PathSim の有効性を示すため，温度や湿度を下げるために選択可能なデバイス数を変化させ，それぞれのデバイス数でのソースコンテキストから対象コンテキストへのデバイス操作系列を導出し，PathSim，SMED，HDS の消費電力量と遷移時間の比較を行った．この実験で使用したデバイスを**表 4**，それぞれのデバイス数で使用したデバイスを**表 5**，実験結果を**図 7** に示す．図 7 は，夏を想定して生成した 12 個のソースコンテキストと 12 個の対象コンテキストに対して，表 5 に示すデバイス数ごとに導出した，12 × 12 の 144 通りのデバイス操作系列の消費電力量と遷移時間それぞれの平均値を表している．また，**表 6** は，デバイス数ごとに SMED および HDS と比べたときの PathSim の消費電力量の削減率を表している．図 7 の棒グラフより，PathSim はデバイス数にかかわらず SMED や HDS より消費電力量が低いことが分かる．また，図 7 の折れ線グラフからデバイス数の増加により遷移時間が短くなることが分かる．表 6 において，デバイス数が 3 のときの SMED に対する削減率が大幅に低下しているのは，エアコンが 2 台になったことで SMED が除湿器を使用しなくなり消費電力量を削減したからである．しかしながら，SMED はそれ以降に同スペックのデバイスが追加されても消費電力量が減ることはない．一方，PathSim は，デバイスが追加されるごとに消費電力量が減少しており，かつ，遷移時間の減少幅が他の手法より大きいことから他の手法と比べてデバイスの追加による効果が大きいといえる．このことから，PathSim はスマートスペースの中で同一タイプのデバイスが複数設置されるような環境での利用が特に有効だといえる．

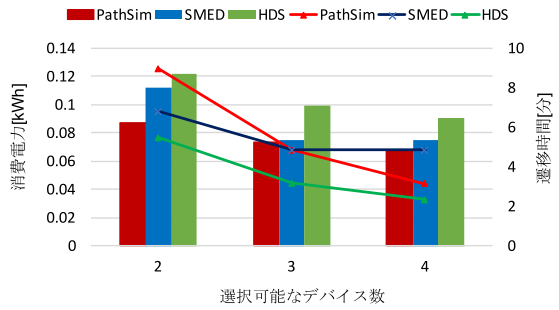

図 7　夏想定時の異なるデバイス数ごとのデバイス操作系列の消費電力量および遷移時間
Fig. 7　Energy consuptions and transition times for each set of appliances during summer simulation.

表 6　異なるデバイス数ごとの PathSim の消費電力量の削減率
Table 6　Comparison of energy consumption for each set of appliances.

	デバイス数			
	2	3	4	平均
SMED	22.4%	1.9%	9.1%	11.1%
HDS	28.4%	26.1%	24.8%	26.5%

6.　まとめ

本論文では，コンテキストアウェアなスマートスペースシステムにおいて，2 つのコンテキスト間の遷移に対する最小コストのデバイス操作系列を導出するツール PathSim を提案した．PathSim は，スマートスペースの変数の定義域を有限個に適切に分割し，(a) コンテキストと関連するイベントを最適に選択/決定し，(b) 選択したイベントを実行したときのコンテキストの遷移と関連する電力消費量をシミュレートし，そして (c) A*アルゴリズムに基づいて最小コストのデバイス操作系列を導出する．日本の夏と冬を想定した計算機シミュレーションにより，PathSim は，空調機器の制御で一般的に使用されているハイブリッドシステムより最大で 50%少ない消費電力量を持つ操作系列を導出できることが分かった．また，PathSim が選択可能な空調デバイスを新たに追加した実験により，デバイス数を増やすことで消費電力量を削減しながら，遷移時間の大幅な短縮が可能であることが分かった．

著者らは文献 [15] において，PathSim を実装したコンテキストアウェアシステムの構成やデバイスの制御方法について提案し，実環境においてシステムの評価実験を行っている．評価実験の結果から，PathSim が導出した操作系列が，実環境においても 10%以下の誤差で目標の温湿度へ遷移できること確認しており，十分実用的であることが分かっている．

今後は，加湿とともに温度が上昇するデバイスや加湿とともに温度の低下するデバイスといったデバイスの副次的効果の考慮，ユーザの好みに合わせられるような目的関

© 2017 Information Processing Society of Japan

数や制約の追加，窓やドアの開閉状況やデバイスの移動といった動的な変化の考慮など PathSim の機能拡張を行う予定である．また，実環境においてユーザ評価を行い，操作性や快適性などの評価を行う予定である．

謝辞 本研究の一部は，JSPS 科研費 16H06980 および 16H01721 の助成によって行った．ここに記して謝意を示す．

参考文献

[1] Weiser, M.: Some computer science issues in ubiquitous computing, *Comm. ACM*, Vol.36, No.7, pp.75–84 (1993).

[2] Henricksena, K. and Indulskab, J.: Developing context-aware pervasive computing applications: Models and approach, *Pervasive and Mobile Computing*, Vol.2, No.1, pp.37–64 (2006).

[3] Baldauf, M., Dustdar, S. and Rosenberg, F.: A survey on context-aware systems, *J. Ad Hoc and Ubiquitous Computing*, Vol.2, No.4, pp.263–277 (2007).

[4] Edwards, W.K. and Grinter, R.E.: At home with ubiquitous computing: Seven challenges, *Proc. UbiComp 2001*, pp.256–272 (2001).

[5] Cook, D.J., Youngblood, M., Heierman III, E.O., Gopalratnam, K., Rao, S., Litvin, A. and Khawaja, F.: MavHome: An agent-based smart home, *Proc. IEEE PerCom 2003*, pp.521–524 (2003).

[6] Nishigaki, K., Yasumoto, K., Shibata, N., Ito, M. and Higashino, T.: Framework and rule-based language for facilitating context-aware computing using information appliances, *Proc. Services and Infrastructure for the Ubiquitous and Mobile Internet* (*SIUMI'05*) (*ICDCS'05 Workshop*), pp.345–351 (2005).

[7] Gu, T., Pung, H.K. and Zhang, D.Q.: A service-oriented middleware for building context-aware services, *J. Network and computer applications*, Vol.28, No.1, pp.1–18 (2005).

[8] Jahn, M., Jentsch, M., Prause, C.R., Pramudianto, F., Al-Akkad, A. and Reiners, R.: The energy aware smart home, *Proc. 5th IEEE Int. Conf. on Future Information Technology* (*FutureTech2010*), pp.1–8 (2010).

[9] Bettini, C., Brdiczka, O., Henricksen, K., Indulska, J., Nicklas, D., Ranganathan, A. and Riboni, D.: A survey of context modelling and reasoning techniques, *J. Pervasive and Mobile Computing*, Vol.6, No.2, pp.161–180 (2010).

[10] Hart, P.E., Nilsson, N.J. and Raphael, B.: A formal basis for the heuristic determination of minimum cost paths, *IEEE Trans. SSC*, Vol.4, No.2, pp.100–107 (1968).

[11] Kashimoto, Y., Ogura, K., Yamamoto, S., Yasumoto, K. and Ito, M.: Saving energy in smart homes with minimal comfort level reduction, *Workshop Proc. IEEE PerCom 2013*, pp.372–376 (2013).

[12] Tetens, O.: Uber einige meteorologische Begriffe, *Zeitschrift fur Geophysik*, Vol.6, pp.297–309 (1930).

[13] 株式会社クレセル：Q & A 快適な温度・湿度は？，入手先 ⟨http://www.crecer.jp/Q-A/HTML/A-11.html⟩ (参照 2014-02-05)．

[14] Henzinger, T.: The theory of hybrid automata, *Proc. 11th Ann. Symp. Logic in Computer Science*, pp.278–292 (1996).

[15] 宵 憲治，Khaled El-Fakih，安本慶一：目標温湿度環境への遷移を最小コストで実現する家電制御システム，情報処理学会論文誌コンシューマ・デバイス&システム，Vol.6, No.1, pp.44–51 (2016).

水本 旭洋 （正会員）

2009 年近畿大学理工学部卒業．2014 年奈良先端科学技術大学院大学情報科学研究科博士後期課程修了．2016 年より同大学博士研究員．博士（工学）．ユビキタスコンピューティングに関する研究に従事．

宵 憲治 （正会員）

2014 年大阪大学基礎工学部情報科学科卒業．2016 年奈良先端科学技術大学院大学情報科学研究科博士前期課程修了．現在，株式会社野村総合研究所に勤務．

カレッド エル=ファキ

経営学の文学士号，コンピュータサイエンスの理学士号および修士号をレバノンアメリカン大学（レバノン共和国）から取得．2002 年コンピュータサイエンスの Ph.D. をオタワ大学（カナダ）から取得．1997 年 Ph.D. フェローとして IBM トロント研究所に勤務．1998 年検証エンジニアとして Cambrian Systems Corporation に勤務．2001 年よりシャルジャアメリカン大学（UAE）助教．現在，同大学准教授．ソフトウェア，通信プロトコル，分散システムのテストに関する研究に従事．

安本 慶一 （正会員）

1991 年大阪大学基礎工学部情報工学科卒業．1995 年同大学大学院博士後期課程退学後，滋賀大学経済学部助手．2002 年より奈良先端科学技術大学院大学情報科学研究科助教授．博士（工学）．現在，同大学同研究科教授．モバイル，ユビキタスコンピューティングに関する研究に従事．ACM，IEEE 各会員．

© 2017 Information Processing Society of Japan