Composing Strand Spaces

Federico Crazzolara * and Glynn Winskel

Computer Laboratory University of Cambridge
{fc232,gw104}@cl.cam.ac.uk

Abstract. The strand space model for the analysis of security protocols is known to have some limitations in the patterns of nondeterminism it allows and in the ways in which strand spaces can be composed. Its successful application to a broad range of security protocols may therefore seem surprising. This paper gives a formal explanation of the wide applicability of strand spaces. We start with an extension of strand spaces which permits several operations to be defined in a compositional way, forming a process language for building up strand spaces. We then show, under reasonable conditions how to reduce the extended strand spaces to ones of the traditional kind. For security protocols we are mainly interested in their safety properties. This suggests a strand-space equivalence: two strand spaces are equivalent if and only if they have essentially the same sets of bundles. However this equivalence is not a congruence with respect to the strand-space operations. By extending the notion of bundle we show how to define the strand-space operations directly on "bundle spaces". This leads to a characterisation of the largest congruence within the strand-space equivalence. Finally, we relate strand spaces to event structures, a well known model for concurrency.

1 Introduction

Security protocols describe a way of exchanging data over an untrusted medium so that, for example, data is not leaked and authentication between the participants in the protocol is guaranteed. The last few years have seen the emergence of successful intensional, event-based, approaches to reasoning about security protocols. The methods are concerned with reasoning about the events that a security protocol can perform, and make use of a causal dependency that exists between events. The method of strand spaces [9–11] has been designed to support such an event-based style of reasoning and has successfully been applied to a broad number of security protocols.

Strand spaces don't compose readily however, not using traditional process operations at least. Their form doesn't allow prefixing by a single event. Nondeterminism only arises through the choice as to where input comes from, and there is not a recognisable nondeterministic sum of strand spaces. Even an easy definition of parallel composition by juxtaposition is thwarted if "unique origination" is handled as a global condition on the entire strand space. That strand

* **BRICS** Centre of the Danish National Research Foundation.

spaces are able to tackle a broad class of security protocols may therefore seem surprising. A reason for the adequacy of strand spaces lies in the fact that they can sidestep conflict if there are enough replicated strands available, which is the case for a broad range of security protocols.

This paper has four main objectives. Firstly it extends the strand space formalism to allow several operations on strand spaces to be defined. The operations form a strand-space language. Secondly the wide applicability of strand spaces to numerous security protocols and properties is backed up formally. The paper documents part of the work done in proving the relation between nets and strand spaces we reported in [3]. Thirdly we address another issue of compositionality. We consider languages of strand-space bundles as models of process behaviour and show how to compose such languages so that they may be used directly in giving the semantics of security protocols. Strand spaces that have substantially the same bundles can be regarded as equivalent and are congruent if they exhibit substantially the same *open* bundles. This congruence lays the ground for equational reasoning between strand spaces. Finally we show how strand spaces relate to event structures.

The results in this paper express the adequacy of strand spaces and relate strand spaces to event structures only with respect to the languages, i.e., sets of finite behaviours they generate. This is not unduly restrictive, however, as in security protocols we are mainly interested in safety properties, properties which stand or fall according to whether they hold for all finite behaviours.

2 Strand spaces

A strand space [11] consists of $\langle s_i \rangle_{i \in I}$, an indexed set of strands. An individual strand s_i, where $i \in I$, is a finite sequence of output or input events carrying output or input actions of the kind $outM$ or inM respectively with M a message built up by encryption and pairing from a set of values (here names) and keys. In the rest of this paper we use n, n_0, n_i to indicate names, A, B, A_0, B_0 to indicate special names which are agent identifiers, and k to stand for a cryptographic key. A name whose first appearance in a strand is on an output message is said to be *originating* on that strand. A name is said to be *uniquely originating* on a strand space if it is originating on only one of its strands.

A strand space has an associated graph whose nodes identify an event of a strand by strand index and position of the event in that strand. Edges are between output and input events concerning the same message and between consecutive events on a same strand. Bundles model protocol runs. A bundle selects those events of a strand space that occur in a run of the protocol and shows the causal dependencies among them which determine the partial order of the events in the run. A bundle is a finite and acyclic subgraph of the strand space graph. Each event in the bundle requires all events that precede it on the same strand (together with the edges that denote the strand precedence). Each input event in the bundle has exactly one incoming edge from an output event.

As an example consider a simplified version of the ISO symmetric key two-pass unilateral authentication protocol (see [2]):

$$\begin{array}{l} A \to B : n \\ B \to A : \{n, A\}_k \quad . \end{array}$$

Agents can engage in a protocol exchange under two different roles, the initiator, here A and the responder, here B. In a protocol round the initiator A chooses a fresh name n and sends it to the responder B. After getting the value, B encrypts it together with the initiator's identifier using a common shared key k. After getting the answer to her challenge, A can decrypt using the shared key and check whether the value sent matches the value received. In that case A can conclude that B is in fact operational.

$$\begin{tikzcd}
\mathit{out}\, n_0 \arrow[r] \arrow[d, Rightarrow] & \mathit{in}\, n_0 \arrow[d, Rightarrow] & \ldots & \mathit{out}\, n_i \arrow[r] \arrow[d, Rightarrow] & \mathit{in}\, n_i \arrow[d, Rightarrow] & \ldots \\
\mathit{in}\, \{n_0, A_0\}_k & \mathit{out}\, \{n_0, A_0\}_k \arrow[l] & & \mathit{in}\, \{n_i, A_0\}_k & \mathit{out}\, \{n_i, A_0\}_k \arrow[l] &
\end{tikzcd}$$

Fig. 1. ISO protocol

The strand-space graph in Figure 1 describes the simple case of only two agents, A_0 and B_0, acting as initiator and responder respectively. For simplicity the graph has been drawn using the actions labelling the events in place of the events themselves. In this simple case if the strand space had a finite number of strands it would itself form a bundle. In the strand space in Figure 1 all names n_i are uniquely originating – there is only one strand on which the first appearance of n_i is in an output action.

Unique origination intends to describe a name as fresh, perhaps chosen at random, and under the assumptions of Dolev and Yao [4], unguessable. For a construction of parallel composition of strand spaces it is therefore reasonable to require that names uniquely originating on components remain so on the composed strand space. Simple juxtaposition of strand spaces does not ensure this. For example, a strand space for the ISO protocol which allows both agents A_0 and B_0 to engage in the protocol in any of the two possible roles – in Figure 2 the strand space formed out of two copies of the one in Figure 1. Figure 3 shows a possible bundle on such strand space. It describes a protocol run with two complete rounds. One in which A_0 is initiator and B_0 responder, and another where the roles are inverted, though the name n_0 is no longer uniquely originating on that strand space. A name's freshness is with respect to a run of a protocol rather than to the whole set of possible executions. A notion of unique origination "in the bundle" seems more appropriate.

Nondeterminism in strand spaces arises only through the choice in a bundle of where input comes from. There is no way of modelling situations in which bundles may be taken either only over one strand space or over another. Juxtaposing

$$
\begin{tikzcd}
out\, n_0 \arrow[r] \arrow[d, Rightarrow] \arrow[dddr, phantom, ""] \arrow[ddr] & in\, n_0 \arrow[d, Rightarrow] & \dots & out\, n_i \arrow[r] \arrow[d, Rightarrow] \arrow[ddr] & in\, n_i \arrow[d, Rightarrow] \\
in\, \{n_0, A_0\}_k & out\, \{n_0, A_0\}_k \arrow[l] & & in\, \{n_i, A_0\}_k & out\, \{n_i, A_0\}_k \arrow[l] \\
out\, n_0 \arrow[r] \arrow[d, Rightarrow] \arrow[uur] & in\, n_0 \arrow[d, Rightarrow] & & out\, n_i \arrow[r] \arrow[d, Rightarrow] \arrow[uur] & in\, n_i \arrow[d, Rightarrow] \\
in\, \{n_0, B_0\}_k & out\, \{n_0, B_0\}_k \arrow[l] & & in\, \{n_i, B_0\}_k & out\, \{n_i, B_0\}_k \arrow[l]
\end{tikzcd}
$$

Fig. 2. ISO protocol - symmetric roles

$$
\begin{tikzcd}
out\, n_0 \arrow[r] \arrow[d, Rightarrow] \arrow[ddr] & in\, n_0 \arrow[d, Rightarrow] \\
in\, \{n_0, A_0\}_k & out\, \{n_0, A_0\}_k \arrow[l] \\
out\, n_0 \arrow[d, Rightarrow] & in\, n_0 \arrow[d, Rightarrow] \\
in\, \{n_0, B_0\}_k & out\, \{n_0, B_0\}_k \arrow[l]
\end{tikzcd}
$$

Fig. 3. A bundle for ISO symmetric roles

strand spaces as we did for example in Figure 2 allows bundles to include events of both components as is the case for the bundle in Figure 3.

One seems to encounter even more difficulties in the attempt to define a construction of prefixing a strand space with an action. Strands can't branch to parallel sub-strands and prefixing each strand with an action would cause as many repetitions of that action as there are strands participating in a bundle. One could add to a strand space a strand containing only the action meant to prefix the strand space. Actions of the "prefixed" strands, however, would not casually depend on that action.

3 Strand spaces with conflict

We extend the definition of strand space, introducing a notion of conflict, which we adapt from event structures [14]. We differ from the original definition of strand spaces in the treatment of unique origination which is taken care of in the definition of bundle rather than being a condition on the entire strand space – the "parametric strand spaces" of [1] achieve a similar effect as to unique origination and are related.

The strands of a strand space are sequences of output and input actions,

$$Act = \{out\, new\, \boldsymbol{n}\, M \mid M \text{ msg., } \boldsymbol{n} \text{ distinct names}\} \cup \{in\, M \mid M \text{ msg.}\} \quad .$$

In $out\, new\, \boldsymbol{n}\, M$, the list $\boldsymbol{n}$ contains distinct names that are intended to be fresh ("uniquely originating") at the event.

Definition 1. A strand space with conflict $(\langle s_i \rangle_{i \in I}, \#)$ consists of:

(i) $\langle s_i \rangle_{i \in I}$ an indexed set of strands with indices I. An individual stand s_i, where $i \in I$, is a finite sequence of output or input actions in Act.
(ii) $\# \subseteq I \times I$ a symmetric, irreflexive binary conflict relation on strand indices.

Strand spaces with an empty conflict relation correspond to those of the standard definition of [11]. We denote by ϵ the empty strand space with no strands and with an empty conflict relation. [1] The empty strand space is different to a strand space $(\langle \lambda \rangle_{i \in I}, \#)$ where each strand is the empty sequence of actions λ. We write $|s|$ for the length of the sequence s.

For a strand space $(\langle s_i \rangle_{i \in I}, \#)$ define the *strand-space graph* $(E, \Rightarrow, \rightarrow, act)$ associated with it as usual (see [11]). It is the graph with nodes (events)

$$E = \{(i, l) \mid i \in I \ , 1 \leq l \leq |s_i|\} \quad ,$$

actions labelling events $act(i, h) = s_i[h]$ and two edge relations. The first expresses precedence among events on the same strand,

$$(i, k) \Rightarrow (i, k+1) \text{ iff } (i, k), (i, k+1) \in E \quad ,$$

and the second expresses all possible communication,

$$(i, l) \rightarrow (j, h) \text{ iff } act(i, l) = out\, new\, \boldsymbol{n}\, M \text{ and } act(j, h) = in\, M \quad .$$

An event is an *input event* if its action is an input action and an event is an *output event* if its action is an output. The names $names(e)$ of an event e are all the names appearing on the action associated with e – the ones that are marked as "new", denoted by $new(e)$, together with those in the message of the action.

Bundles of a strand space describe runs in a computation.

Definition 2. A bundle b of a strand space with conflict $\#$ is a finite subgraph $(E_b, \Rightarrow_b, \rightarrow_b, act_b)$ of the strand-space graph $(E, \Rightarrow, \rightarrow, act)$ such that:

(i) if $e \Rightarrow e'$ and $e' \in E_b$ then $e \Rightarrow_b e'$, *(control precedence)*
(ii) if $e \in E_b$ and $act_b(e) = in\, M$ then there exists a unique $e' \in E_b$ such that $e' \rightarrow_b e$, *(output-input precedence)*
(iii) if $e, e' \in E_b$ such that $act_b(e) = out\, new\, \boldsymbol{n}\, M$ and $n \in \boldsymbol{n} \cap names(e')$ then either $e \Rightarrow_b^* e'$ or there exists an input event e'' such that $n \in names(e'')$ and $e'' \Rightarrow_b^* e'$, *(freshness)*
(iv) if $(i, h), (j, k) \in E_b$ then $\neg(i \# j)$, *(conflict freeness)*
(v) the relation $\Rightarrow_b \cup \rightarrow_b$ is acyclic. *(acyclicity)*

The empty graph, also denoted by λ, is a bundle. It will be clear from the context whether λ stands for the empty bundle or whether it denotes the empty sequence of actions. The empty strand space does not have any bundles.

Points (i), (ii), (v) of the definition of bundle for a strand space with conflict match the standard definition of [11]. Point (iii) ensures freshness of "new" values in a bundle. Point (iv) doesn't allow events from conflicting strands to appear in a bundle. Write $\leq_b$ for the reflexive and transitive closure of $\Rightarrow_b \cup \rightarrow_b$.

---

[1] We won't make much use of this particular strand space; it is however the identity for the operations of parallel composition and nondeterministic sum of strand spaces.

Proposition 1. *If b is a bundle then $\leq_b$ is a partial order on E_b.*

The relation $\leq_b$ determines the partial order on events occurring in a computation described by b. Names introduced as "new" don't appear on events preceding their introduction and are never introduced as "new" more than once.

Proposition 2. *Let b be a bundle of a strand space and let $e, e' \in E_b$ such that $act_b(e) = out\,new\,\boldsymbol{n}\,M$. If $n \in \boldsymbol{n} \cap names(e')$ then $e \leq_b e'$ and if $act_b(e') = out\,new\,\boldsymbol{m}\,M'$ then $n \notin \boldsymbol{m}$.*

We regard two strand spaces as substantially the same if they differ only on the indices of their strands and so that one strand space can be obtained from the other by a simple "re-indexing" operation. [2]

Definition 3. Given $(\langle s_i \rangle_{i \in I}, \#)$ and $(\langle t_j \rangle_{j \in J}, \#')$, two strand spaces, write

$$(\langle s_i \rangle_{i \in I}, \#) \cong (\langle t_j \rangle_{j \in J}, \#')$$

if there exists a bijection $\pi : I \to J$ such that $s_i = t_{\pi(i)}$ for all $i \in I$ and $i \# j$ iff $\pi(i) \#' \pi(j)$ for all $i, j \in I$.

The relation $\cong$ is an equivalence. A bijection π that establishes it is called a *re-indexing of strand spaces.* Let $(\langle s_i \rangle_{i \in I}, \#)$ be a strand space, J be a set, and $\pi : I \to J$ be a bijection. Define the strand space $(\langle t_j \rangle_{j \in J}, \pi(\#))$ where $t_j = s_{\pi^{-1}(j)}$ for all $j \in J$ and $j \, \pi(\#) \, j'$ iff $\pi^{-1}(j) \# \pi^{-1}(j')$ for all $j, j' \in J$. The relation $\pi(\#)$ is irreflexive and symmetric and $(\langle s_{\pi(i)} \rangle_{i \in I}, \pi(\#)) \cong (\langle s_i \rangle_{i \in I}, \#)$.

Proposition 3. *Let $(\langle s_i \rangle_{i \in I}, \#)$ and $(\langle t_j \rangle_{j \in J}, \#')$ be two strand spaces such that $(\langle s_i \rangle_{i \in I}, \#) \cong (\langle t_j \rangle_{j \in J}, \#')$ for a bijection $\pi : I \to J$. If b is a bundle of $(\langle s_i \rangle_{i \in I}, \#)$ then $\pi(b)$ obtained from b by changing all strand indices according to π is a bundle of $(\langle t_j \rangle_{j \in J}, \#')$.*

4 Constructions on strand spaces

Prefixing a strand space with an action is complicated by the strand-space formalism not permitting strands to branch. Only if the strand space to be prefixed is such that every two different strands are in conflict can each strand be prefixed with the action. Then the conflict relation disallows repetitions of the prefixing action in bundles. Given $\alpha \in Act$ and a strand space $(\langle s_i \rangle_{i \in I}, \#)$ such that for all $i, j \in I$ if $i \neq j$ then $i \# j$, define

$$\alpha.(\langle s_i \rangle_{i \in I}, \#) \stackrel{def}{=} (\langle \alpha s_i \rangle_{i \in I}, \#) \quad .$$

We understand the special case of prefixing the empty strand space with an action, to yield the empty strand space $\alpha.\epsilon = \epsilon$. When prefixing a strand space consisting of only empty strands one obtains $\alpha.(\langle \lambda \rangle_{i \in I}, \#) = (\langle \alpha \rangle_{i \in I}, \#)$.

[2] If the indices carry structure (some might involve agent names for example) one might refine the permissible re-indexings.

The *parallel composition* of strand spaces is the disjoint union of their sets of strands and conflict relations. Disjoint union is achieved by tagging each space with a different index. Given a collection of strand spaces $(\langle s_i^k\rangle_{i\in I_k}, \#^k)$ indexed by k in a set K, define

$$||_{k\in K}(\langle s_i^k\rangle_{i\in I_k}, \#^k) \stackrel{def}{=} (\langle s_h\rangle_{h\in H}, \#)$$

where $H = \sum_{k\in K} I_k$, $s_{(k,i)} = s_i^k$, and where $(k,i)\,\#\,(k',i')$ iff $k = k'$ and $i \#^k i'$. Two strands are in conflict only if they belong to the same component of the parallel composition and are in conflict within that component. In particular if K is the empty set then the parallel composition yields ϵ.

As a special case of parallel composition of strand spaces consider the strand space obtained by composing infinitely many identical strand spaces. Abbreviate

$$!\,(\langle s_i\rangle_{i\in I}, \#) \stackrel{def}{=} ||_{k\in\omega}(\langle s_i\rangle_{i\in I}, \#) \quad .$$

One observes that $!\,(\langle s_i\rangle_{i\in I}, \#) = (\langle s_{(n,i)}\rangle_{(n,i)\in\omega\times I}, !\#)$ where $!\#$ is the binary relation over $\omega \times I$ such that $(n,i)\,!\#\,(m,i')$ iff $n = m$ and $i\,\#\,i'$.

The *nondeterministic sum* of strand spaces constructs the same indexed set of strands as the operation of parallel composition. The conflict relation of a summed space however, in addition to the existing conflicts, imposes conflict between every two strands that belong to different components. Given a collection of strand spaces $(\langle s_i^k\rangle_{i\in I_k}, \#^k)$ indexed by k in a set K, define

$$\sum_{k\in K} S_k \stackrel{def}{=} (\langle s_h\rangle_{h\in H}, \#)$$

where $(k,i)\,\#\,(k',i')$ iff either $k \neq k'$ or ($k = k'$ and $i \#^k i'$). Two strands are in conflict only if they belong to different components or are already in conflict within a component. The relation $\#$ is irreflexive and symmetric. The indexing set H and the strands remain the same as for parallel composition.

5 A process language for strand spaces

The language $\mathcal{S}$ of strand spaces has the following grammar:

$$S \ ::= \ L \mid \sum_{j\in J} S_j \mid ||_{j\in J} S_j$$

where $L \in \mathcal{L}$, the language of "sequential strand spaces" is given by

$$L \ ::= \ \langle\lambda\rangle \mid \alpha.L \mid \sum_{j\in J} L_j \quad .$$

The strand space $\langle\lambda\rangle$ has only one strand which is the empty sequence of actions and has empty conflict. [3] The bundles of strand spaces in $\mathcal{L}$ form linearly ordered sets of events, and therefore can be thought of as runs of a sequential process.

[3] Let the index of the empty strand in $\langle\lambda\rangle$ be a distinguished index $*$.

A strand-space term of $\mathcal{S}$ is a "par" process – parallel composition is only at the top level and so the term consists of a parallel composition and sum of sequential processes. The "!-par" processes are those terms of $\mathcal{S}$ of the form $!S$.

6 Open bundles

The usual semantics of a strand space is expressed in terms of its set of bundles. By broadening to open bundles, the open-bundle space can be built compositionally. An *open bundle* has the structure of a bundle where, however, input events need not necessarily be related to output events. An open bundle is "open" to the environment for communication on input events that are not linked to output events.

Definition 4. An open bundle b of a strand space with conflict $\#$ is a finite subgraph $(E_b, \Rightarrow_b, \rightarrow_b, act_b)$ of the strand-space graph $(E, \Rightarrow, \rightarrow, act)$ such that:

(i) if $e \Rightarrow_G e'$ and $e' \in E_b$ then $e \Rightarrow_b e'$, *(control precedence)*
(ii) if $e' \rightarrow_b e$ and $e'' \rightarrow_b e$ then $e' = e''$, *(output-input correspondence)*
(iii) if $e, e' \in E_b$ s.t. $act(e) = out\,new\boldsymbol{n}M$ and $n \in \boldsymbol{n} \cap names(e')$ then either $e \Rightarrow_b^* e'$ or there exists an input event $e'' \in E_b$ such that $n \in names(e'')$, and $e'' \Rightarrow_b^* e'$, *(freshness)*
(iv) if $(i,h), (j,k) \in E_b$ then $\neg(i \# j)$, *(conflict freeness)*
(v) the relation $\Rightarrow_b \cup \rightarrow_b \cup \hookrightarrow$ is acyclic, where $e \hookrightarrow e'$ if $e \neq e'$ and if $new(e) \cap names(e') \neq \emptyset$. *(acyclicity)*

In an open bundle "freshness" dependencies need not be caught through $\Rightarrow_b$ and $\rightarrow_b$. Point (v) takes account of additional freshness dependencies and excludes graphs like

$$\begin{tikzcd} in\,n \arrow[d, Rightarrow] & in\,m \arrow[d, Rightarrow] \\ out\,new\,m\ m \arrow[ur] & out\,new\,n\ n \end{tikzcd} .$$

Our constructions for open bundles take us through the intermediary of control graphs (which are similar to pomsets [7] and message sequence charts [5]).

A *control graph*, with indices I, is a graph $(E, \rightarrow, act)$ where $E \subseteq I \times \mathbb{N}$ such that if $(i,h) \in E$ and $h > 1$ then $(i, h-1) \in E$ (when we write $(i, h-1) \Rightarrow (i,h)$), and where $\rightarrow \subseteq E \times E$, and $act : E \rightarrow Act$. Denote by $ind(g)$ the set of indices of a control graph g. Strand-space graphs, bundles and open bundles are examples of control graphs. We say a control graph is an open bundle when it is finite and satisfies axioms (ii), (iii) and (v) above.

Prefixing extends a control graph with a new initial control node $(i,1)$ where i is an index. Every event in the original graph that has index i is shifted one position later in the control structure while keeping its casual dependencies. For i, j indices and $h \in \mathbb{N}$, define $(j,h)/i = (j, h+1)$ if $j = i$ and $(j,h)/i = (j,h)$ otherwise. For an index i, $\alpha \in Act$ and g a control graph, define the control graph

$$(\alpha, i) \,.\, g = (E, \rightarrow, act)$$

where $E = \{(i,1)\} \cup \{e/i \mid e \in E_g\}$ and $e/i \to e'/i$ whenever $e \to_g e'$. Take $act(i,1) = \alpha$ and $act(e/i) = act_g(e)$ for every $e \in E_g$.

Control graphs can be *composed* by tagging the number of components to keep them disjoint and juxtaposing them. For i, j indices and $h \in w$, define $j : (i,h) = ((j,i),h)$. For a control graph g and index j define $j : g = (E, \to, act)$ where $E = \{j : e \mid e \in E_g\}$ with $j : e \to j : e'$ whenever $e \to_g e'$ and actions $act(j : e) = act_g(e)$.

The definition of open bundle ensures that b extends to a bundle over a (bigger) strand space. Let g, g' be control graphs. Define $g \preceq g'$ iff $E_g = E_{g'}$, $\to_g \subseteq \to_{g'}$, and $act_g = act_{g'}$. Write $g{\uparrow} = \{b \mid g \preceq b \text{ and } b \text{ an open bundle}\}$.

Lemma 1. *Let b be an open bundle of a strand space S. There exists a strand space T and an open bundle t of T such that among the open bundles in $(t||b){\uparrow}$ there is a bundle of $T||S$.*

The language of open bundles of a strand-space term is defined as follows:

Definition 5. Let $L \in \mathcal{L}$ and $S \in \mathcal{S}$. Define

$$
\begin{aligned}
\mathcal{O}(\langle\lambda\rangle) &= \{\lambda\} \\
\mathcal{O}(\alpha.L) &= \{\lambda\} \;\cup\; \{(\alpha,i)\,.\,\lambda \mid i \in ind(L)\} \;\cup \\
&\qquad \bigcup\{(\alpha,i)\,.\,b{\uparrow} \;\mid b \in \mathcal{O}(L) \setminus \{\lambda\} \;\&\; i \in ind(b)\}
\end{aligned}
$$

$$\mathcal{O}(\sum_{j\in J} S_j) = \{j : b \mid b \in \mathcal{O}(S_j)\}$$

$$\mathcal{O}(||_{j\in J} S_j) = \bigcup\{(\bigcup_{i\in I} i : b_i){\uparrow} \;\mid I \subseteq J \;\&\; I \text{ finite } \&\; \forall i \in I\,.\, b_i \in \mathcal{O}(S_i)\} \quad .$$

Theorem 1. *If S is a strand-space term in $\mathcal{S}$ then the elements of $\mathcal{O}(S)$ are exactly the open bundles of the strand space denoted by S.*

7 Strand space equivalence

We have seen an equivalence relation that relates two strand spaces if, via re-indexing, they become the same space. That relation of isomorphism, as expected, is respected by the operations on strand-spaces. When security properties are expressed as safety properties on the language of bundles of a strand space one doesn't, however, want to distinguish between strand spaces that have isomorphic bundle languages.

Let b, b' be two bundles. Write $b \cong b'$ iff there exists a bijection $\phi : E_b \to E_{b'}$ such that

(i) if $e \to_b e'$ then $\phi(e) \to_{b'} \phi(e')$,
(ii) if $e \Rightarrow_b e'$ then $\phi(e) \Rightarrow_{b'} \phi(e')$,
(iii) $act_b(e) = act_{b'}(\phi(e))$.

Definition 6. Let $S, S' \in \mathcal{S}$. Define $\approx$ the symmetric relation such that $S \approx S'$ iff for every bundle b of S there exists a bundle b' of S' such that $b \cong b'$.

Proposition 4. *The relation $\approx$ is an equivalence relation.*

The equivalence relation $\approx$ is not a congruence. For example, the strand-space terms $in\,M.\epsilon$ and $in\,N.\epsilon$ where N and M are two different messages are in the relation $\approx$ – they both have only one bundle, λ. A distinguishing context exists:

$$in\,M.\epsilon \mid\mid out\,M.\epsilon \not\approx in\,N.\epsilon \mid\mid out\,M.\epsilon \quad .$$

The parallel composition on the left hand side has the bundle $in\,M \longleftarrow out\,M$, that on the right only λ.

A context for a strand-space term in the language $\mathcal{S}$ is defined as follows:

$$C ::= [\;] \mid \alpha.C \mid ||_{i\in I}T_i \mid \Sigma_{i\in I}T_i$$

where for each context of the form $||_{i\in I}T_i$ or $\Sigma_{i\in I}T_i$ there is exactly one $i \in I$ such that T_i is a context C and $T_j \in \mathcal{S}$ for all $j \in I\backslash\{i\}$. The context $[\;]$ is a placeholder for a strand-space term. Write $C[S]$ for the term obtained by replacing the strand-space term S for $[\;]$ in context C in the obvious way. An equivalence relation on strand-space terms is a congruence if it respects all contexts.

Definition 7. Let $S, S' \in \mathcal{S}$. Define $\approx_{\mathcal{O}}$ the smallest symmetric relation such that $S \approx_{\mathcal{O}} S'$ if for every open bundle b of S there exists an open bundle b' of S' such that $b \cong b'$.

Theorem 2. *The relation $\approx_{\mathcal{O}}$ is the largest congruence relation inside $\approx$.*

8 Eliminating conflict

If one doesn't restrict the number of rounds of a protocol an agent can do one can hope to model the protocol with a strand space of the form $!\,(\langle s_i\rangle_{i\in I}, \#)$. In that case a simpler model, obtained by dropping the conflict relation, exhibits substantially the same behaviour as the more complex strand space with conflict.

Lemma 2. *Given I a set of indices, a finite set $A \subseteq \omega \times I$, and $\#$ a conflict relation over I then there exists a bijection $\pi : \omega \times I \to \omega \times I$ where for all $(n,i) \in \omega \times I$ there exists $m \in \omega$ such that $\pi(n,i) = (m,i)$ and $\neg(\pi(u) \;!\#\; \pi(v))$ for all $u, v \in A$.*

The previous lemma and the observation that two different copies of the same strand space have the same strands at corresponding positions, yield:

Theorem 3. *Consider strand spaces $!\,(\langle s_i\rangle_{i\in I}, \emptyset)$ and $!\,(\langle s_i\rangle_{i\in I}, \#)$. Let b be a bundle of $!\,(\langle s_i\rangle_{i\in I}, \emptyset)$. There exists a strand space S such that b is a bundle of S and $S \cong !\,(\langle s_i\rangle_{i\in I}, \#)$.*

The behaviour of a replicated strand space with conflict corresponds, modulo re-indexing, to that of the strand space one obtains by dropping the conflict relation.

Corollary 1. *Consider strand spaces* $!(\langle s_i \rangle_{i \in I}, \emptyset)$ *and* $!(\langle s_i \rangle_{i \in I}, \#)$ *strand spaces.*

(i) If b *is bundle of* $!\,(\langle s_i \rangle_{i \in I}, \#)$ *then* b *is bundle of* $!\,(\langle s_i \rangle_{i \in I}, \emptyset)$.
(ii) If b *is bundle of* $!\,(\langle s_i \rangle_{i \in I}, \emptyset)$ *then there exists a re-indexing* π *such that* $\pi(b)$ *is a bundle of* $!\,(\langle s_i \rangle_{i \in I}, \#)$.

9 Event structures from strand spaces

A bundle is a graph and therefore a set of edges and nodes. It turns out that the bundles of a strand space ordered by inclusion correspond to the finite configurations of a prime event structure. Prime event structures are a particularly simple kind of event structure where the enabling of events can be expressed as a global partial order of causal dependency [6, 12].

Definition 8. A prime event structure $E = (E, \#, \leq)$ consists of a set E of events partially ordered by the causal dependency relation $\leq$ and a binary, symmetric, irreflexive conflict relation $\# \subseteq E \times E$, which satisfy

(i) $\{e' \mid e' \leq e\}$ is finite for all $e \in E$, and
(ii) if $e \# e' \leq e''$ then $e \# e''$ for all $e, e', e'' \in E$.

Definition 9. The configurations of $(E, \#, \leq)$ consist of $x \subseteq E$ such that

(i) $\forall e, e' \in x \,.\, \neg(e \# e')$ and *(conflict free)*
(ii) $\forall e, e' . e' \leq e \in x \Rightarrow e' \in x$. *(left closed)*

Write $\mathcal{F}(E)$ for the set of configurations of an event structure and $\mathcal{F}^{fin}(E)$ for its finite configurations. Let $\mathcal{B}$ be the set of bundles of a strand space. A subset X of $\mathcal{B}$ is *compatible* iff there exists $b \in \mathcal{B}$ such that $b' \subseteq b$ for all $b' \in X$. We say X is pairwise compatible if every subset of X with two elements is compatible.

Proposition 5. *The partial order* $(\mathcal{B}, \subseteq)$ *satisfies the following properties:*

(i) if $X \subseteq \mathcal{B}$, X *is finite and pairwise compatible, then* $\bigcup X \in \mathcal{B}$, *(coherence)*
(ii) if $X \subseteq \mathcal{B}$ *and* X *is compatible, then* $\bigcap X \in \mathcal{B}$. *(stability)*

Given $b \in \mathcal{B}$ and $e \in E_b$ define $\lceil e \rceil_b \stackrel{def}{=} \bigcap \{b' \in \mathcal{B} \mid e \in b' \ \wedge\ b' \subseteq b\}$.

Proposition 6. *For every* $b \in \mathcal{B}$ *and every* $e \in E_b$ *the set* $\lceil e \rceil_b$ *is a bundle in* $\mathcal{B}$. *For every finite and compatible* $X \subseteq \mathcal{B}$ *if* $\lceil e \rceil_b \subseteq \bigcup X$ *then* $\exists b' \in X \,.\, \lceil e \rceil_b \subseteq b'$.

We call a bundle $\lceil e \rceil_b$ a prime. The primes form a basis for $(\mathcal{B}, \subseteq)$.

Proposition 7. *Every* $b \in \mathcal{B}$ *can be obtained as* $b = \bigcup \{p \mid p \subseteq b,\ p \text{ prime}\}$.

Consider the structure $Pr(\mathcal{B}) \stackrel{def}{=} (P, \#, \subseteq)$ where P is the set of primes of $\mathcal{B}$ and where $p \# p'$ iff the two primes p and p' are not compatible.

Theorem 4. *The structure $Pr(\mathcal{B})$ is a prime event structure.*

Theorem 5. *$\phi : (\mathcal{B}, \subseteq) \cong (\mathcal{F}^{fin}Pr(\mathcal{B}), \subseteq)$ such that $\phi(b) = \{p \mid p \subseteq b,\ p\ prime\}$ is an isomorphism of partial orders with inverse map $\theta : \mathcal{F}^{fin}Pr(\mathcal{B}) \to \mathcal{B}$ given by $\theta(x) = \bigcup x$.*

Since $Pr(\mathcal{B})$ is a prime event structure the partial order $(\mathcal{F}Pr(\mathcal{B}), \subseteq)$ is a special Scott domain of information, a prime algebraic domain (see [6, 12–14]). This domain, however, can include configurations which are infinite and therefore are not bundles in the usual sense. The prime event structures $Pr(\mathcal{B})$ underlie the strand-space semantics Syverson gave to the BAN logic [8].

Concluding remarks. This paper exposes the central position within the theory of strand spaces of the congruence $\approx_{\mathcal{O}}$, based on having the same open bundles. It suggests building an equational theory around $\approx_{\mathcal{O}}$. Its usefulness in supporting abstract specifications would seem to require an operation to hide (or internalise) events in addition to those of Section 5.

References

1. I. Cervesato, N. A. Durgin, P. D. Lincoln, J. C. Mitchell, and A. Scedrov. Relating strands and multiset rewriting for security protocol analysis. *13th CSFW*, 2000.
2. J. Clark and J. Jacob. A survey of authentication protocol literature: V. 1.0. 1997.
3. F. Crazzolara and G. Winskel. Events in security protocols. *8th ACM CCS*, 2001.
4. D. Dolev and A. C. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 2(29), 1983.
5. ITU-TS. *ITU-TS Recommendation Z.120: Message Sequence Charts (MSC)*. 1997.
6. M. Nielsen, G. Plotkin, and G. Winskel. Petri nets, Event structures and Domains. *Theoretical Computer Science*, 13, 1981.
7. V. R. Pratt. Modelling concurrency with partial orders. *International Journal of Parallel Programming*, 15(1):33–71, 1986.
8. P. Syverson. Towards a Strand Semantics for Authentication Logic. *Electronic Notes in Theoretical Computer Science*, (20), 1999.
9. J. Thayer and J. Guttman. Authentication tests. *IEEE Symposium on Security and Privacy*, 2000.
10. J. Thayer, J. Herzog, and J. Guttman. Honest ideals on strand spaces. *11th CSFW*, 1998.
11. J. Thayer, J. Herzog, and J. Guttman. Strand spaces: Why is a security protocol correct? *IEEE Symposium on Security and Privacy*, 1998.
12. G. Winskel. *Events in computation*. PhD thesis, University of Edinburgh, 1980.
13. G. Winskel. Event structures. *Adv. Course on Petri nets*, vol. 255 of *LNCS*, 1987.
14. G. Winskel. An introduction to event structures. *Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency*, vol. 354 of *LNCS*, 1988.