Semantics and Verification of Software

Lecture 20: Dataflow Analysis III (The Framework)

Thomas Noll

Lehrstuhl für Informatik 2
(Software Modeling and Verification)

RWTH Aachen University

noll@cs.rwth-aachen.de

http://www-i2.informatik.rwth-aachen.de/i2/svsw10/

Summer Semester 2010

Outline

Similarities between Analysis Problems

- **Observation:** the analyses presented so far have some similarities

$\Longrightarrow$ Look for underlying framework

- **Advantage:** possibility for designing (efficient) generic algorithms for solving dataflow equations
- **Overall pattern:** for $c \in Cmd$ and $l \in L_c$, the analysis information (AI) is described by equations of the form

$$\mathsf{AI}_l = \begin{cases} \iota & \text{if } l \in E \\ \bigsqcup\{\varphi_{l'}(\mathsf{AI}_{l'}) \mid (l', l) \in F\} & \text{otherwise} \end{cases}$$

where

- the set of extremal labels, E, is $\{\mathsf{init}(c)\}$ or $\mathsf{final}(c)$
- ι specifies the extremal analysis information
- the combination operator, $\bigsqcup$, is $\bigcap$ or $\bigcup$
- $\varphi_{l'}$ denotes the transfer function of block $B^{l'}$
- the flow relation F is $\mathsf{flow}(c)$ or $\mathsf{flow}^R(c)$ $(:= \{(l', l) \mid (l, l') \in \mathsf{flow}(c)\})$

Characterization of Analyses

- **Direction of information flow:**
  - forward:
    - $F = \mathsf{flow}(c)$
    - AI_l concerns entry of B^l
    - c has isolated entry
  - backward:
    - $F = \mathsf{flow}^R(c)$
    - AI_l concerns exit of B^l
    - c has isolated exits
- **Quantification over paths:**
  - may:
    - $\bigsqcup = \bigcup$
    - property satisfied by some path
    - interested in least solution (later)
  - must:
    - $\bigsqcup = \bigcap$
    - property satisfied by all paths
    - interested in greatest solution (later)

Roadmap

Goal: solve dataflow equation system by fixpoint iteration

1. Introduce partial order for comparing analysis results
2. Establish least upper bound as combination operator
3. Ensure monotonicity of transfer functions
4. Guarantee termination of fixpoint iteration (and continuity of functional) by Ascending Chain Condition
5. Optimize fixpoint iteration by worklist algorithm

Outline

Partial Orders

The domain of analysis information usually forms a partial order where the ordering relation compares the "precision" of information.

Definition 20.1 (Partial order; repetition of Def. 7.1)

A partial order (PO) $(D, \sqsubseteq)$ consists of a set D, called domain, and of a relation $\sqsubseteq \subseteq D \times D$ such that, for every $d_1, d_2, d_3 \in D$,

reflexivity: $d_1 \sqsubseteq d_1$

transitivity: $d_1 \sqsubseteq d_2$ and $d_2 \sqsubseteq d_3 \implies d_1 \sqsubseteq d_3$

antisymmetry: $d_1 \sqsubseteq d_2$ and $d_2 \sqsubseteq d_1 \implies d_1 = d_2$

It is called total if, in addition, always $d_1 \sqsubseteq d_2$ or $d_2 \sqsubseteq d_1$.

Partial Orders

The domain of analysis information usually forms a partial order where the ordering relation compares the “precision” of information.

Definition 20.1 (Partial order; repetition of Def. 7.1)

A partial order (PO) $(D, \sqsubseteq)$ consists of a set D, called domain, and of a relation $\sqsubseteq \subseteq D \times D$ such that, for every $d_1, d_2, d_3 \in D$,

reflexivity: $d_1 \sqsubseteq d_1$

transitivity: $d_1 \sqsubseteq d_2$ and $d_2 \sqsubseteq d_3 \implies d_1 \sqsubseteq d_3$

antisymmetry: $d_1 \sqsubseteq d_2$ and $d_2 \sqsubseteq d_1 \implies d_1 = d_2$

It is called total if, in addition, always $d_1 \sqsubseteq d_2$ or $d_2 \sqsubseteq d_1$.

Example 20.2

1. (Live Variables) $(2^{Var_c}, \subseteq)$ is a (non-total) partial order

Partial Orders

The domain of analysis information usually forms a partial order where the ordering relation compares the "precision" of information.

Definition 20.1 (Partial order; repetition of Def. 7.1)

A partial order (PO) $(D, \sqsubseteq)$ consists of a set D, called domain, and of a relation $\sqsubseteq \subseteq D \times D$ such that, for every $d_1, d_2, d_3 \in D$,

reflexivity: $d_1 \sqsubseteq d_1$

transitivity: $d_1 \sqsubseteq d_2$ and $d_2 \sqsubseteq d_3 \implies d_1 \sqsubseteq d_3$

antisymmetry: $d_1 \sqsubseteq d_2$ and $d_2 \sqsubseteq d_1 \implies d_1 = d_2$

It is called total if, in addition, always $d_1 \sqsubseteq d_2$ or $d_2 \sqsubseteq d_1$.

Example 20.2

1. (Live Variables) $(2^{Var_c}, \subseteq)$ is a (non-total) partial order
2. (Available Expressions) $(2^{AExp_c}, \supseteq)$ is a (non-total) partial order

Upper Bounds

In the dataflow equation system, analysis information from several predecessors is combined by taking the least upper bound.

Definition 20.3 ((Least) upper bound; repetition of Def. 7.4)

Let $(D, \sqsubseteq)$ be a partial order and $S \subseteq D$.

1. An element $d \in D$ is called an upper bound of S if $s \sqsubseteq d$ for every $s \in S$ (notation: $S \sqsubseteq d$).

Upper Bounds

In the dataflow equation system, analysis information from several predecessors is combined by taking the least upper bound.

Definition 20.3 ((Least) upper bound; repetition of Def. 7.4)

Let $(D, \sqsubseteq)$ be a partial order and $S \subseteq D$.

1. An element $d \in D$ is called an upper bound of S if $s \sqsubseteq d$ for every $s \in S$ (notation: $S \sqsubseteq d$).
2. An upper bound d of S is called least upper bound (LUB) or supremum of S if $d \sqsubseteq d'$ for every upper bound d' of S (notation: $d = \bigsqcup S$).

Upper Bounds

In the dataflow equation system, analysis information from several predecessors is combined by taking the least upper bound.

Definition 20.3 ((Least) upper bound; repetition of Def. 7.4)

Let $(D, \sqsubseteq)$ be a partial order and $S \subseteq D$.

1. An element $d \in D$ is called an upper bound of S if $s \sqsubseteq d$ for every $s \in S$ (notation: $S \sqsubseteq d$).
2. An upper bound d of S is called least upper bound (LUB) or supremum of S if $d \sqsubseteq d'$ for every upper bound d' of S (notation: $d = \bigsqcup S$).

Example 20.4

1. (Live Variables) $(D, \sqsubseteq) = (2^{Var_c}, \subseteq)$. Given $V_1, \ldots, V_n \subseteq Var_c$,
$$\bigsqcup\{V_1, \ldots, V_n\} = \bigcup\{V_1, \ldots, V_n\}$$

Upper Bounds

In the dataflow equation system, analysis information from several predecessors is combined by taking the least upper bound.

Definition 20.3 ((Least) upper bound; repetition of Def. 7.4)

Let $(D, \sqsubseteq)$ be a partial order and $S \subseteq D$.

1. An element $d \in D$ is called an upper bound of S if $s \sqsubseteq d$ for every $s \in S$ (notation: $S \sqsubseteq d$).
2. An upper bound d of S is called least upper bound (LUB) or supremum of S if $d \sqsubseteq d'$ for every upper bound d' of S (notation: $d = \bigsqcup S$).

Example 20.4

1. (Live Variables) $(D, \sqsubseteq) = (2^{Var_c}, \subseteq)$. Given $V_1, \ldots, V_n \subseteq Var_c$,
$$\bigsqcup\{V_1, \ldots, V_n\} = \bigcup\{V_1, \ldots, V_n\}$$
2. (Avail. Expr.) $(D, \sqsubseteq) = (2^{AExp_c}, \supseteq)$. Given $A_1, \ldots, A_n \subseteq AExp_c$,
$$\bigsqcup\{A_1, \ldots, A_n\} = \bigcap\{A_1, \ldots, A_n\}$$

Complete Lattices

Since $\{\varphi_{l'}(\mathsf{Al}_{l'}) \mid (l', l) \in F\}$ is not necessarily a chain (Def. 7.4), chain completeness (Def. 7.6) is not sufficient for guaranteeing the well-definedness of the equation system. A stronger property is required:

Definition 20.5 (Complete lattice)

A complete lattice is a partial order $(D, \sqsubseteq)$ such that all subsets of D have least upper bounds. In this case,

$$\bot := \bigsqcup \emptyset$$

denotes the least element of D.

Complete Lattices

Since $\{\varphi_{l'}(\mathsf{Al}_{l'}) \mid (l', l) \in F\}$ is not necessarily a chain (Def. 7.4), chain completeness (Def. 7.6) is not sufficient for guaranteeing the well-definedness of the equation system. A stronger property is required:

Definition 20.5 (Complete lattice)

A complete lattice is a partial order $(D, \sqsubseteq)$ such that all subsets of D have least upper bounds. In this case,

$$\bot := \bigsqcup \emptyset$$

denotes the least element of D.

Example 20.6

1. (Live Variables)
   $(D, \sqsubseteq) = (2^{Var_c}, \subseteq)$ is a complete lattice with $\bot = \emptyset$

Complete Lattices

Since $\{\varphi_{l'}(\mathsf{Al}_{l'}) \mid (l', l) \in F\}$ is not necessarily a chain (Def. 7.4), chain completeness (Def. 7.6) is not sufficient for guaranteeing the well-definedness of the equation system. A stronger property is required:

Definition 20.5 (Complete lattice)

A complete lattice is a partial order $(D, \sqsubseteq)$ such that all subsets of D have least upper bounds. In this case,

$$\bot := \bigsqcup \emptyset$$

denotes the least element of D.

Example 20.6

1. (Live Variables)
   $(D, \sqsubseteq) = (2^{Var_c}, \subseteq)$ is a complete lattice with $\bot = \emptyset$
2. (Available Expressions)
   $(D, \sqsubseteq) = (2^{AExp_c}, \supseteq)$ is a complete lattice with $\bot = AExp_c$

Duality in Complete Lattices

- Dual concept of least upper bound: greatest lower bound
- **Definitions:**
  - An element $d \in D$ is called a lower bound of $S \subseteq D$ if $d \sqsubseteq s$ for every $s \in S$ (notation: $d \sqsubseteq S$).
  - A lower bound d is called greatest lower bound (GLB) or infimum of S if $d' \sqsubseteq d$ for every lower bound d' of S (notation: $d = \bigsqcap S$).

Duality in Complete Lattices

- Dual concept of least upper bound: greatest lower bound
- **Definitions:**
  - An element $d \in D$ is called a lower bound of $S \subseteq D$ if $d \sqsubseteq s$ for every $s \in S$ (notation: $d \sqsubseteq S$).
  - A lower bound d is called greatest lower bound (GLB) or infimum of S if $d' \sqsubseteq d$ for every lower bound d' of S (notation: $d = \bigsqcap S$).
- **Examples:**
  - (Live Variables) $(D, \sqsubseteq) = (2^{Var_c}, \subseteq)$, $\bigsqcap\{V_1, \ldots, V_n\} = \bigcap\{V_1, \ldots, V_n\}$
  - (Available Expressions) $(D, \sqsubseteq) = (2^{AExp_c}, \supseteq)$, $\bigsqcap\{A_1, \ldots, A_n\} = \bigcup\{A_1, \ldots, A_n\}$

Duality in Complete Lattices

- Dual concept of least upper bound: greatest lower bound
- **Definitions:**
  - An element $d \in D$ is called a lower bound of $S \subseteq D$ if $d \sqsubseteq s$ for every $s \in S$ (notation: $d \sqsubseteq S$).
  - A lower bound d is called greatest lower bound (GLB) or infimum of S if $d' \sqsubseteq d$ for every lower bound d' of S (notation: $d = \bigsqcap S$).
- **Examples:**
  - (Live Variables) $(D, \sqsubseteq) = (2^{Var_c}, \subseteq)$, $\bigsqcap\{V_1, \ldots, V_n\} = \bigcap\{V_1, \ldots, V_n\}$
  - (Available Expressions) $(D, \sqsubseteq) = (2^{AExp_c}, \supseteq)$, $\bigsqcap\{A_1, \ldots, A_n\} = \bigcup\{A_1, \ldots, A_n\}$
- **Lemma:** the following are equivalent:
  - $(D, \sqsubseteq)$ is a complete lattice (i.e., every subset of D has a least upper bound)
  - Every subset of D has a greatest lower bound

Duality in Complete Lattices

- Dual concept of least upper bound: greatest lower bound
- **Definitions:**
  - An element $d \in D$ is called a lower bound of $S \subseteq D$ if $d \sqsubseteq s$ for every $s \in S$ (notation: $d \sqsubseteq S$).
  - A lower bound d is called greatest lower bound (GLB) or infimum of S if $d' \sqsubseteq d$ for every lower bound d' of S (notation: $d = \bigsqcap S$).
- **Examples:**
  - (Live Variables) $(D, \sqsubseteq) = (2^{Var_c}, \subseteq)$, $\bigsqcap\{V_1, \ldots, V_n\} = \bigcap\{V_1, \ldots, V_n\}$
  - (Available Expressions) $(D, \sqsubseteq) = (2^{AExp_c}, \supseteq)$, $\bigsqcap\{A_1, \ldots, A_n\} = \bigcup\{A_1, \ldots, A_n\}$
- **Lemma:** the following are equivalent:
  - $(D, \sqsubseteq)$ is a complete lattice (i.e., every subset of D has a least upper bound)
  - Every subset of D has a greatest lower bound
- **Corollary:** every complete lattice has a greatest element $\top := \bigsqcap \emptyset$

Chains

Chains are generated by the approximation of the analysis information in the fixpoint iteration.

Definition 20.7 (Chain; repetition of Def. 7.4 and 7.6)

Let $(D, \sqsubseteq)$ be a partial order.

1. A subset $S \subseteq D$ is called a chain in D if, for every $s_1, s_2 \in S$,
$$s_1 \sqsubseteq s_2 \text{ or } s_2 \sqsubseteq s_1$$
(that is, S is a totally ordered subset of D).

Chains

Chains are generated by the approximation of the analysis information in the fixpoint iteration.

Definition 20.7 (Chain; repetition of Def. 7.4 and 7.6)

Let $(D, \sqsubseteq)$ be a partial order.

1. A subset $S \subseteq D$ is called a chain in D if, for every $s_1, s_2 \in S$,
$$s_1 \sqsubseteq s_2 \text{ or } s_2 \sqsubseteq s_1$$
(that is, S is a totally ordered subset of D).
2. $(D, \sqsubseteq)$ is called chain complete (CCPO) if each of its chains has a least upper bound.

Chains

Chains are generated by the approximation of the analysis information in the fixpoint iteration.

Definition 20.7 (Chain; repetition of Def. 7.4 and 7.6)

Let $(D, \sqsubseteq)$ be a partial order.

1. A subset $S \subseteq D$ is called a chain in D if, for every $s_1, s_2 \in S$,
$$s_1 \sqsubseteq s_2 \text{ or } s_2 \sqsubseteq s_1$$
(that is, S is a totally ordered subset of D).
2. $(D, \sqsubseteq)$ is called chain complete (CCPO) if each of its chains has a least upper bound.

Corollary 20.8

Every complete lattice is a CCPO.

Monotonicity of Functions

The monotonicity of transfer functions (which formalize the impact of a block in the program on the analysis information) excludes "oscillating behavior" in fixpoint iteration.

Definition 20.9 (Monotonicity; repetition of Def. 8.1)

Let $(D, \sqsubseteq)$ and $(D', \sqsubseteq')$ be partial orders, and let $F : D \to D'$. F is called monotonic (w.r.t. $(D, \sqsubseteq)$ and $(D', \sqsubseteq')$) if, for every $d_1, d_2 \in D$,

$$d_1 \sqsubseteq d_2 \implies F(d_1) \sqsubseteq' F(d_2).$$

Monotonicity of Functions

The monotonicity of transfer functions (which formalize the impact of a block in the program on the analysis information) excludes "oscillating behavior" in fixpoint iteration.

Definition 20.9 (Monotonicity; repetition of Def. 8.1)

Let $(D, \sqsubseteq)$ and $(D', \sqsubseteq')$ be partial orders, and let $F : D \to D'$. F is called monotonic (w.r.t. $(D, \sqsubseteq)$ and $(D', \sqsubseteq')$) if, for every $d_1, d_2 \in D$,

$$d_1 \sqsubseteq d_2 \implies F(d_1) \sqsubseteq' F(d_2).$$

Example 20.10

1. (Live Variables) $(D, \sqsubseteq) = (2^{Var_c}, \subseteq)$
   Each transfer function $\varphi_{l'}(V) := (V \setminus \mathsf{kill}_{\mathsf{LV}}(B^{l'})) \cup \mathsf{gen}_{\mathsf{LV}}(B^{l'})$ is obviously monotonic

Monotonicity of Functions

The monotonicity of transfer functions (which formalize the impact of a block in the program on the analysis information) excludes "oscillating behavior" in fixpoint iteration.

Definition 20.9 (Monotonicity; repetition of Def. 8.1)

Let $(D, \sqsubseteq)$ and $(D', \sqsubseteq')$ be partial orders, and let $F : D \to D'$. F is called monotonic (w.r.t. $(D, \sqsubseteq)$ and $(D', \sqsubseteq')$) if, for every $d_1, d_2 \in D$,

$$d_1 \sqsubseteq d_2 \implies F(d_1) \sqsubseteq' F(d_2).$$

Example 20.10

1. (Live Variables) $(D, \sqsubseteq) = (2^{Var_c}, \subseteq)$
   Each transfer function $\varphi_{l'}(V) := (V \setminus \mathsf{kill}_{\mathsf{LV}}(B^{l'})) \cup \mathsf{gen}_{\mathsf{LV}}(B^{l'})$ is obviously monotonic
2. (Available Expressions) $(D, \sqsubseteq) = (2^{AExp_c}, \supseteq)$
   Each transfer function $\varphi_{l'}(A) := (A \setminus \mathsf{kill}_{\mathsf{AE}}(B^{l'})) \cup \mathsf{gen}_{\mathsf{AE}}(B^{l'})$ is obviously monotonic

The Ascending Chain Condition

Termination of fixpoint iteration is guaranteed by the Ascending Chain Condition.

Definition 20.11 (Ascending Chain Condition)

A partial order $(D, \sqsubseteq)$ satisfies the Ascending Chain Condition (ACC) if each ascending chain $d_1 \sqsubseteq d_2 \sqsubseteq \ldots$ eventually stabilizes, i.e., there exists $n \in \mathbb{N}$ such that $d_n = d_{n+1} = \ldots$

The Ascending Chain Condition

Termination of fixpoint iteration is guaranteed by the Ascending Chain Condition.

Definition 20.11 (Ascending Chain Condition)

A partial order $(D, \sqsubseteq)$ satisfies the Ascending Chain Condition (ACC) if each ascending chain $d_1 \sqsubseteq d_2 \sqsubseteq \ldots$ eventually stabilizes, i.e., there exists $n \in \mathbb{N}$ such that $d_n = d_{n+1} = \ldots$

Example 20.12

1. (Live Variables) $(D, \sqsubseteq) = (2^{Var_c}, \subseteq)$ satisfies ACC since Var_c (unlike Var) is finite

The Ascending Chain Condition

Termination of fixpoint iteration is guaranteed by the Ascending Chain Condition.

Definition 20.11 (Ascending Chain Condition)

A partial order $(D, \sqsubseteq)$ satisfies the Ascending Chain Condition (ACC) if each ascending chain $d_1 \sqsubseteq d_2 \sqsubseteq \ldots$ eventually stabilizes, i.e., there exists $n \in \mathbb{N}$ such that $d_n = d_{n+1} = \ldots$

Example 20.12

1. (Live Variables) $(D, \sqsubseteq) = (2^{Var_c}, \subseteq)$ satisfies ACC since Var_c (unlike Var) is finite
2. (Available Expressions) $(D, \sqsubseteq) = (2^{AExp_c}, \supseteq)$ satisfies ACC since $AExp_c$ (unlike $AExp$) is finite

Fixpoints

Theorem 20.13 (Fixpoint Theorem; repetition of Thm. 8.7)

Let $(D, \sqsubseteq)$ be a CCPO and $F : D \to D$ continuous. Then

$$\mathsf{fix}(F) := \bigsqcup \{F^n(\bigsqcup \emptyset) \mid n \in \mathbb{N}\}$$

is the least fixpoint of F.

Fixpoints

Theorem 20.13 (Fixpoint Theorem; repetition of Thm. 8.7)

Let $(D, \sqsubseteq)$ be a CCPO and $F : D \to D$ continuous. Then

$$\mathsf{fix}(F) := \bigsqcup \{F^n (\bigsqcup \emptyset) \mid n \in \mathbb{N}\}$$

is the least fixpoint of F.

Definition 20.14 (Continuity; repetition of Def. 8.5)

Let $(D, \sqsubseteq)$ and $(D', \sqsubseteq')$ be CCPOs and $F : D \to D'$ monotonic. Then F is called continuous (w.r.t. $(D, \sqsubseteq)$ and $(D', \sqsubseteq')$) if, for every non-empty chain $S \subseteq D$,

$$F \left(\bigsqcup S\right) = \bigsqcup F(S).$$

Fixpoints

Theorem 20.13 (Fixpoint Theorem; repetition of Thm. 8.7)

Let $(D, \sqsubseteq)$ be a CCPO and $F : D \to D$ continuous. Then

$$\mathsf{fix}(F) := \bigsqcup \{F^n (\bigsqcup \emptyset) \mid n \in \mathbb{N}\}$$

is the least fixpoint of F.

Definition 20.14 (Continuity; repetition of Def. 8.5)

Let $(D, \sqsubseteq)$ and $(D', \sqsubseteq')$ be CCPOs and $F : D \to D'$ monotonic. Then F is called continuous (w.r.t. $(D, \sqsubseteq)$ and $(D', \sqsubseteq')$) if, for every non-empty chain $S \subseteq D$,

$$F\left(\bigsqcup S\right) = \bigsqcup F(S).$$

Corollary 20.15

Monotonic functions on partial orders that satisfy ACC are continuous.

Fixpoints

Theorem 20.13 (Fixpoint Theorem; repetition of Thm. 8.7)

Let $(D, \sqsubseteq)$ be a CCPO and $F : D \to D$ continuous. Then

$$\mathsf{fix}(F) := \bigsqcup \{F^n \left(\bigsqcup \emptyset\right) \mid n \in \mathbb{N}\}$$

is the least fixpoint of F.

Definition 20.14 (Continuity; repetition of Def. 8.5)

Let $(D, \sqsubseteq)$ and $(D', \sqsubseteq')$ be CCPOs and $F : D \to D'$ monotonic. Then F is called continuous (w.r.t. $(D, \sqsubseteq)$ and $(D', \sqsubseteq')$) if, for every non-empty chain $S \subseteq D$,

$$F\left(\bigsqcup S\right) = \bigsqcup F(S).$$

Corollary 20.15

Monotonic functions on partial orders that satisfy ACC are continuous.

Proof.

on the board □

Outline

Dataflow Systems I

Definition 20.16 (Dataflow system)

A dataflow system $S = (L, E, F, (D, \sqsubseteq), \iota, \varphi)$ consists of

- a finite set of (program) labels L (here: L_c),
- a set of extremal labels $E \subseteq L$ (here: $\{\mathsf{init}(c)\}$ or $\mathsf{final}(c)$),
- a flow relation $F \subseteq L \times L$ (here: $\mathsf{flow}(c)$ or $\mathsf{flow}^R(c)$),
- a complete lattice $(D, \sqsubseteq)$ that satisfies ACC (with LUB operator $\bigsqcup$ and least element $\bot$),
- an extremal value $\iota \in D$ (for the extremal labels), and
- a collection of monotonic transfer functions $\{\varphi_l \mid l \in L\}$ of type $\varphi_l : D \to D$.

Example 20.17

Problem	**Available Expressions**	**Live Variables**
E	$\{\mathsf{init}(c)\}$	$\mathsf{final}(c)$
F	$\mathsf{flow}(c)$	$\mathsf{flow}^R(c)$
D	2^{AExp_c}	2^{Var_c}
$\sqsubseteq$	$\supseteq$	$\subseteq$
$\bigsqcup$	$\bigcap$	$\bigcup$
$\perp$	$AExp_c$	$\emptyset$
ι	$\emptyset$	Var_c
φ_l	$\varphi_l(d) = (d \setminus \mathsf{kill}(B^l)) \cup \mathsf{gen}(B^l)$	