

Homework 3

Instructor: YiZheng Zhao*Name:* 张运吉, *StudentId:* 211300063**Question 1. Closure under Disjoint Union**

Let C be an \mathcal{ALC} -Concept. If there exists a rule that can be applied, then C has a subconcept of the form $\neg D$, and D is as the form of $E \sqcap F, E \sqcup F, \neg E, \exists r.E, \forall r.E$. Let C' be the \mathcal{ALC} -concept after rule application.

- case 1: $\neg D = \neg(E \sqcap F)$, E, F is concept name.

After transforming, $\neg(E \sqcap F)$ became $\neg\neg\neg E \sqcup \neg\neg\neg F$.

$$\begin{aligned}
 M(\neg(E \sqcap F)) &= \{\#(E \sqcap F)\} \cup M(E) \cup M(F) \\
 M(\neg\neg\neg E \sqcup \neg\neg\neg F) &= \{\#(\neg\neg E), \#(\neg E), \#E, \#(\neg\neg F), \#(\neg F), \#F\} \\
 &\quad \cup M(E) \cup M(F) \\
 M(C') &= (M(C) \setminus M(\neg D)) \cup M(\neg\neg\neg E \sqcup \neg\neg\neg F) \\
 &= (M(C) \setminus \{\#(E \sqcap F)\}) \\
 &\quad \cup \{\#(\neg\neg E), \#(\neg E), \#E, \#(\neg\neg F), \#(\neg F), \#F\}
 \end{aligned}$$

So, we can get that after transforming, $\#(E \sqcap F)$ was replaced by some smaller numbers $\#(\neg\neg E), \#(\neg E), \#E, \#(\neg\neg F), \#(\neg F), \#F$.

- case 2: $\neg D = \neg(E \sqcup F)$, E, F is concept name.

The prove of case 2 is similar as case 1.

- case 3: $\neg D = \neg\neg E$, E is a concept name.

After transforming, $\neg\neg E$ became E .

$$M(\neg\neg E) = \{\# \neg E, \# E\}$$

$$M(C') = (M(C) \setminus M(\neg\neg E)) \cup M(E) = M(C) \setminus \{\# \neg E, \# E\}$$

So, we can get that after transforming, $\# \neg E, \# E$ was deleted.

- case 4: $\neg D = \neg(\exists r.E)$, E is a concept name.

$$M(\neg(\exists r.E)) = \{\#(\exists r.E)\} \cup M(E)$$

$$M(\forall r.\neg E) = \{\#E\} \cup M(E)$$

$$M(C') = (M(C) \setminus M(\neg(\exists r.E))) \cup M(\forall r.\neg E) = (M(C) \setminus \{\#(\exists r.E)\}) \cup \{\#E\}$$

So, we can get that after transforming, $\#(\exists r.E)$ was replaced by a smaller number $\#E$.

- case 5: $\neg D = \neg(\forall r.E)$, E is a concept name.

The prove of case 5 is similar as case 4.

Finally, we can get that with transforming, the elements of $M(C)$ will became smaller over and over again, the stop condition is the elements of $M(C)$ all becomes 0. The procedure of transformations is always terminates because the numbers are limited.

Question 2. Negation Normal Norm (NNF)

(1) \implies :

let \mathcal{T}' is obtained from \mathcal{T} by replacing each concept definition $A \equiv C$ with the concept inclusion $A \sqsubseteq C$ and $C \sqsubseteq A$, \mathcal{T} is equivalent to \mathcal{T}' .

Obviously, $\mathcal{T}^\sqsubseteq \subseteq \mathcal{T}'$, which means any model of $\subseteq \mathcal{T}'$ is also a model of \mathcal{T}^\sqsubseteq .

So, every concept name is satisfiable w.r.t. \mathcal{T} implies it is satisfiable w.r.t. \mathcal{T}^\sqsubseteq .

\Leftarrow :

If concept name C is satisfiable w.r.t. \mathcal{T}^\sqsubseteq , then there exists an interpretation \mathcal{I} s.t. $C^\mathcal{I} \neq \emptyset$ w.r.t. \mathcal{T}^\sqsubseteq .

We expand \mathcal{I} and get \mathcal{J} . The rule to expand is as follow: We modify recursively all $A^\mathcal{J}, C_1^\mathcal{J}, C_2^\mathcal{J}, \dots, C_n^\mathcal{J}$ to $A^\mathcal{I} \cup C_1^\mathcal{I} \cup C_2^\mathcal{I} \cup \dots \cup C_n^\mathcal{I}$ if GCI $A \sqsubseteq C_i$ in \mathcal{T}^\sqsubseteq for all $1 \leq i \leq n$, until all $A^\mathcal{J} = C^\mathcal{J}$ if concept definition $A \equiv C$ in \mathcal{T} .

Because the stop condition is $A^\mathcal{J} = C^\mathcal{J}$ and the expand procedure does not violate any one GCI, so \mathcal{J} is a interpretation w.r.t. \mathcal{T} . Because we just expand some $A^\mathcal{I}$ and get corresponding $A^\mathcal{J}$, so $C^\mathcal{J} \neq \emptyset$ due to $C^\mathcal{I} \neq \emptyset$.

Therefore, C is satisfiable w.r.t. \mathcal{T} .

(2) Do not holds.

$$\begin{aligned}\mathcal{T} &= \{A \equiv C \sqcap \neg B, B \equiv P, C \equiv P\} \\ \mathcal{T}^\sqsubseteq &= \{A \sqsubseteq C \sqcap \neg B, B \sqsubseteq P, C \sqsubseteq P\}\end{aligned}$$

Because:

$$A^\mathcal{I} = (C \sqcap \neg B)^\mathcal{I} = C^\mathcal{I} \cap (\Delta^\mathcal{I} \setminus B^\mathcal{I}) = P^\mathcal{I} \cap (\Delta^\mathcal{I} \setminus P^\mathcal{I}) = \emptyset$$

So concept name \mathcal{A} is not satisfiable w.r.t. \mathcal{T} .

Let $\Delta^\mathcal{I} = \{a\}, A^\mathcal{I} = \{a\}, C^\mathcal{I} = \{a\}, B^\mathcal{I} = \emptyset, P^\mathcal{I} = \{a\}$, obviously it satisfies \mathcal{T}^\sqsubseteq and $A^\mathcal{I} \neq \emptyset$.

So concept name A is satisfiable w.r.t. \mathcal{T}^\sqsubseteq .

Question 3. Tableau Algorithm for ABoxes with Acyclic TBoxes

- Termination.

We had known that the tableau algorithm $\text{consistent}(\mathcal{A})$ is terminate, so we only need to prove that after using \equiv_1 -rule and \equiv_2 -rule to unfold \mathcal{T} , the new ABox \mathcal{A}' is finite. Obviously, we would add finite number of assertions to \mathcal{A} , so the new ABox \mathcal{A}' is finite.

- Soundness.

Let $\mathcal{A}' = \text{consistent}(\mathcal{K}, \mathcal{A})$. To construct a model from a complete and clash-free ABox \mathcal{A}' , we can use the same definitions as presented in the lecture slides to obtain an interpretation \mathcal{I} of all role names and of the concept names that do not have definitions in \mathcal{T} . It remains to show that \mathcal{I} is also a model of \mathcal{A}' , where the main problem is showing Property (P1) by induction. So let's prove it.

If C doesn't have definition in \mathcal{T} , then we have already proved it. Otherwise C has definition. According to the \equiv_1 -rule and \equiv_2 -rule, if $a : A$ and $A \equiv C$, there must be $a : C$ in \mathcal{A}' , if $a : \neg A$ and $A \equiv C$, there must be $a : \neg C$, by the definition of \mathcal{I} , $a^{\mathcal{I}} \in A^{\mathcal{I}}$.

- Completeness.

The \equiv_1 -rule: if $a : A \in \mathcal{A}$, $A \equiv C \in \mathcal{T}$, then $a^{\mathcal{I}} \in A^{\mathcal{I}}$, $A^{\mathcal{I}} = C^{\mathcal{I}}$, so $a^{\mathcal{I}} \in C^{\mathcal{I}}$. Therefore, \mathcal{I} is still a model of $\mathcal{A} \cup \{a : C\}$.

The \equiv_2 -rule: if $a : \neg A \in \mathcal{A}$, $A \equiv C \in \mathcal{T}$, then $a^{\mathcal{I}} \notin A^{\mathcal{I}}$, $A^{\mathcal{I}} = C^{\mathcal{I}}$, so $a^{\mathcal{I}} \notin C^{\mathcal{I}}$. Therefore, \mathcal{I} is still a model of $\mathcal{A} \cup \{a : \neg C\}$.

The other rules are as the same as presented in the text book.

Therefore, $\text{consistent}(\mathcal{T}, \mathcal{A})$ is a decision procedure for the consistency of \mathcal{ALC} -knowledge bases with acyclic TBoxes.

Question 4. Tableau Algorithm for ABoxes with Acyclic TBoxes

Doesn't hold.

If the subsumption $\neg(\forall r.A) \sqcap \forall r.C \sqsubseteq \forall r.E$ doesn't hold w.r.t. acyclic TBox \mathcal{T} , then $\neg(\forall r.A) \sqcap \forall r.C \sqcap \neg(\forall r.E)$ holds w.r.t. acyclic TBox \mathcal{T} .

We use tableau algorithm to determine it.

$$\begin{aligned}
\mathcal{A}_0 &= \{a : \neg(\forall r.A) \sqcap \forall r.C \sqcap \neg(\forall r.E)\} \\
\mathcal{A}_1 &= \mathcal{A}_0 \cup \{a : \neg(\forall r.A), a : \forall r.C, a : \neg(\forall r.E)\} \\
&= \mathcal{A}_0 \cup \{a : \exists r.\neg A, a : \forall r.C, a : \exists r.\neg E\} \quad (\sqcap\text{-rule}) \\
\mathcal{A}_2 &= \mathcal{A}_1 \cup \{(a, b) : r, b : \neg A, (a, c) : r, c : \neg E\} \quad (\exists\text{-rule}) \\
\mathcal{A}_3 &= \mathcal{A}_2 \cup \{b : C, c : C\} \quad (\forall\text{-rule}) \\
\mathcal{A}_4 &= \mathcal{A}_3 \cup \{b : (\exists r.\neg B) \sqcap \neg A, c : (\exists r.\neg B) \sqcap \neg A\} \quad (\equiv_1\text{-rule}) \\
\mathcal{A}_5 &= \mathcal{A}_4 \cup \{c : \exists r.A \sqcup \forall r.\neg D\} \quad (\equiv_2\text{-rule}) \\
\mathcal{A}_6 &= \mathcal{A}_5 \cup \{b : \exists r.\neg B, c : \exists r.\neg B, c : \neg A\} \quad (\sqcap\text{-rule}) \\
\mathcal{A}_7 &= \mathcal{A}_6 \cup \{(b, d) : r, d : \neg B, (c, e) : r, e : \neg B\} \quad (\exists\text{-rule}) \\
\mathcal{A}_8 &= \mathcal{A}_7 \cup \{c : \exists r.A\} \quad (\sqcup\text{-rule}) \\
\mathcal{A}_9 &= \mathcal{A}_8 \cup \{(c, f) : r, f : A\} \quad (\exists\text{-rule})
\end{aligned}$$

There is no rules to apply in \mathcal{A}_9 and it is clash free, which means $\neg(\forall r.A) \sqcap \forall r.C \sqcap \neg(\forall r.E)$ holds w.r.t. acyclic TBox \mathcal{T} and $\neg(\forall r.A) \sqcap \forall r.C \sqsubseteq \forall r.E$ doesn't hold w.r.t. acyclic TBox \mathcal{T}

Question 5. Anywhere Blocking

- Termination.

Let $m = |\text{sub}(K)|$. Termination is a consequence of the following properties of the expansion rules:

- (1) There can be at most $|\text{sub}(\mathcal{K})|$ rule applications of a individual name.
 - (2) The outdegree of each tree in the forest-shaped *ABox* is bounded by $|\text{sub}(\mathcal{K})|$.
 - (3) Any path having more individual names than $2^{|\text{sub}(\mathcal{K})|}$ has at least two individual names a, b such that $\text{con}_{\mathcal{A}}(b) = \text{con}_{\mathcal{A}}(a) \subseteq \text{sub}(\mathcal{K})$. Because the ages of a individual name are strictly monotonic increasing, we assume that $\text{age}(a) < \text{age}(b)$. Therefore, there are at most 2^m individual names in a path along tree individuals that are not blocked. That means the depth of each tree in the forest-shaped *ABox* is bounded by 2^m .
- Soundness.

Let $\mathcal{A}' = \text{consistent}(\mathcal{K})$. We firstly construct \mathcal{A}' :

$$\begin{aligned} \mathcal{A}'' = & \{a : C \mid a : C \in \mathcal{A}' \text{ and } a \text{ is not blocked} \} \cup \\ & \{(a, b) : r \mid (a, b) : r \in \mathcal{A}' \text{ and } b \text{ is not blocked} \} \cup \\ & \{(a, b') : r \mid (a, b) : r \in \mathcal{A}', a \text{ is not blocked and } b \text{ is blocked by } b'\} \end{aligned}$$

The first preconclusion we want to show is:

$$\text{con}_{\mathcal{A}''}(a) = \text{con}_{\mathcal{A}'}(a)$$

- For individual assertion, a must not be blocked by its definition.
- For role assertion, there is two case:
 - case 1: $(a, b) \in r \in \mathcal{A}'$, because the successor of a blocked individual is also blocked, so a must not be blocked.
 - case 2: $(a, b') \notin r \in \mathcal{A}'$, b' is not blocked according to the definition of anywhere blocking.

Therefore, there is no blocked individual names in \mathcal{A}'' , which means the first preconclusion holds.

The second preconclusion we want to show is:

\mathcal{A}'' is complete if \mathcal{A}' is complete.

- \sqcap – rule: if $a : C \sqcap D \in \mathcal{A}''$, according to the first preconclusion $a : C \sqcap D \in \mathcal{A}'$. Completeness of \mathcal{A}' implies that $\{a : C, a : D\} \subseteq \mathcal{A}'$ and then the first preconclusion implies $\{a : C, a : D\} \subseteq \mathcal{A}''$.
- \sqcup – rule: if $a : C \sqcup D \in \mathcal{A}''$, according to the first preconclusion $a : C \sqcup D \in \mathcal{A}'$. Completeness of \mathcal{A}' implies that at least one of $\{a : C, a : D\}$ is in \mathcal{A}' and then the first preconclusion implies at least one of $\{a : C, a : D\}$ is in \mathcal{A}'' .
- \sqsubseteq – rule: if $C \sqsubseteq D \in \mathcal{T}$ and $a : C \in \mathcal{A}''$, according to the first preconclusion $a : C \sqsubseteq D \in \mathcal{A}'$. Completeness of \mathcal{A}' implies that $a : D \in \mathcal{A}'$ and then the first preconclusion implies $a : D \in \mathcal{A}''$.
- \exists – rule: if $a : \exists r.C \in \mathcal{A}''$, according to the first preconclusion $a : \exists r.C \in \mathcal{A}'$ and a is not blocked in \mathcal{A}' . Completeness of \mathcal{A}' implies that $\{(a, b) : r, b : C\} \subseteq \mathcal{A}'$. If b is not blocked, $\{(a, b) : r, b : C\} \subseteq \mathcal{A}''$ according to the definition of \mathcal{A}'' . If b is anywhere blocked by b' , then b' is not blocked and $\text{con}_{\mathcal{A}'}(b) \subseteq \text{con}_{\mathcal{A}'}(b')$ according to the definition of anywhere blocking. Therefore, $b' : C \in \mathcal{A}''$ according to the first preconclusion. By definition of \mathcal{A}'' , we have $(a, b') : r \in \mathcal{A}''$ naturally.
- \forall –rule: if $\{a : \forall r.C, (a, b') : r\} \subseteq \mathcal{A}''$, then $a : \forall r.C \in \mathcal{A}'$ and neither a or b' not blocked in \mathcal{A}' . If $(a, b') : r \in \mathcal{A}'$, then completeness of \mathcal{A}' implies $b' : C \in \mathcal{A}''$. If $(a, b') : r \notin \mathcal{A}'$, then there must be a b such that $(a, b) : r \in \mathcal{A}'$ and b is anywhere blocked by b' in \mathcal{A}' . Then completeness of \mathcal{A}' implies $b : C \in \mathcal{A}'$. According to the definition of anywhere blocking, we

have $\text{con}_{\mathcal{A}'}(b) \subseteq \text{con}_{\mathcal{A}'}(b')$, and $b' : C \in \mathcal{A}'$. Then the first preconclusion implies that $b' : C \in \mathcal{A}''$.

Now, we can construct a interpretation \mathcal{I} according to \mathcal{A}'' just like the prove of lemma 4.5 in the text book and then we can get that \mathcal{I} is a model of \mathcal{A}'' and \mathcal{A} .

For each GCI $C \sqsubseteq D \in \mathcal{T}$ and assertion $a : C \in \mathcal{A}$, we have $a : C \in \mathcal{A}''$ and $a : D \in \mathcal{A}''$. If $a^{\mathcal{I}} \in C^{\mathcal{I}}$, then $a^{\mathcal{I}} \in D^{\mathcal{I}}$ and thus $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. Therefore, \mathcal{I} is a model of \mathcal{T} .

To sum up, \mathcal{I} is a model of $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, \mathcal{K} is consistent if the tableau algorithm with anywhere blocking returns "consistent".

- Completeness.

The prove of completeness is similar as the prove on the text book. The only difference is \sqsubseteq -rule: If $a : C \in \mathcal{A}$ and $C \sqsubseteq D \in \mathcal{T}$, then $a^{\mathcal{I}} \in C^{\mathcal{I}}$ and $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ by semantics. Consequently, adding $a : D$ to \mathcal{A} has no effect on the consistency of \mathcal{K} . Therefore, \mathcal{I} is still a model of $(\mathcal{T}, \mathcal{A} \cup \{a : D\})$.

Question 6. Precompletion of Tableau Algorithm

\implies :

Let \mathcal{J} denote a model of \mathcal{K} . According to the completeness of tableau algorithm, applying any rule to \mathcal{A} will not change the consistency of \mathcal{K} for any model with respect to \mathcal{K} . Therefore, \mathcal{J} is also a model with respect to \mathcal{A}' (precompletion of \mathcal{K}).

By the definition of $C_{\mathcal{A}}^a$, obviously $a^{\mathcal{J}} \in C_{\mathcal{A}'}^a$. So $C_{\mathcal{A}'}^a$ is satisfiable with respect to \mathcal{T} for each individual name a that occurs in \mathcal{A}' .

\impliedby :

We construct a suitable interpretation \mathcal{I} :

- $\Delta^I = \{a \mid a : C \in \mathcal{A}'\}$
- $a^I = a$ for all individual name a in \mathcal{A}'
- $A^I = \{a \mid A \in \text{con}_{\mathcal{A}'}(a)\}$ for each $A \in \text{sub}(\mathcal{A}')$
- $r^I = \{(a, b) \mid (a, b) : r \in \mathcal{A}'\}$ for each role name r in \mathcal{A}'

If $C_{\mathcal{A}}^a$ is satisfiable for all individual names a in \mathcal{A}' , then there exists at least one individual name a such that $a^I \in C_{\mathcal{A}'}^a = (\sqcap_{a:C \in \mathcal{A}'} C)^I = \bigcap_{a:C \in \mathcal{A}'} C^I$, which means $a^I \in C^I$ for all individual assertion $a : C \in \mathcal{A}'$. Therefore, \mathcal{A}' is consistent with the model \mathcal{I} .

To show that \mathcal{T} is satisfied by \mathcal{I} , we consider each GCI $C \sqsubseteq D \in \mathcal{T}$ and each assertion $a : C \in \mathcal{A}$. Since $a : C \in \mathcal{A}$, we have $a : C \in \mathcal{A}'$, and therefore $a : D \in \mathcal{A}'$ by the completeness of \mathcal{A}' . Thus $a^I \in C^I$ implies $a^I \in D^I$ and thus we have $C^I \subseteq D^I$. That means \mathcal{I} is a model of \mathcal{T} .

Hence \mathcal{I} is a model of $\mathcal{K} = (\mathcal{T}, \mathcal{A})$. So, \mathcal{K} is consistent.

Question 7. Tableau Algorithm for \mathcal{ALCN}

- Soundness.

For soundness, the proof is similar to the one for Lemma 4.11 (in the text book), but the construction of \mathcal{A}'' must be modified so that it leads to a complete and clash-free ABox. For example, if $\{a : (\geq 2r), (a, x) : r, (a, y) : r, x \neq y\} \subseteq \mathcal{A}'$, and both x and y are blocked by z , then replacing $(a, x) : r$ and $(a, y) : r$ with $(a, z) : r$ in \mathcal{A}'' would effectively merge x and y into z , resulting in a clash. So we extend \mathcal{A}' by adding some copies of blocking individual name for each individual name they block. For example, x is blocked by z , we add a new individual name z_1 and concept assertion $z_1 : C$ for all $z : C$ and role assertions $(z_1, y) : r$ for all $(z, y) : r$. Trivially, our new \mathcal{A}'' is still complete and clash-free because any rule could be applied to it would have applied to its original individual name. Now we could prove the

soundness using the known method, except when x is blocked by z , we treat it as it is blocked by z_1 . Finally, we can copy the equality assertions from \mathcal{A}' to \mathcal{A}'' and use these in the model construction to ensure that each individual name occurring in \mathcal{A} is appropriately interpreted.

- Completeness.
 - The \geq -rule: If $a : (\geq n r) \in \mathcal{A}$, then $a^{\mathcal{J}} \in (\geq n r)^{\mathcal{J}}$. We can extend \mathcal{J} to get a new model \mathcal{J}' of \mathcal{A} , we just add n new individual names $d_i, 1 \leq i \leq n$ and $(a^{\mathcal{J}'}, d_i^{\mathcal{J}'}) \in (\geq nr)^{\mathcal{J}'}$, so $\mathcal{A} \cup \{(a, d_i) : r, d_i : C \mid 1 \leq i \leq n\} \cup \{d_i \neq d_j \mid 1 \leq i < j \leq n\}$ is still consistent.
 - The \leq -rule: If $a : (\leq n r) \in \mathcal{A}$, then $a^{\mathcal{J}} \in (\leq n r)^{\mathcal{J}}$, which means there at most n distingusih individual names d such that $(a^{\mathcal{J}}, d^{\mathcal{J}})$. Therefore, at least one $\mathcal{A}[b_j \mapsto b_i] \cup \{b_i = b_j\}$ is consistent.

Question 8. Tableau Algorithm for \mathcal{ALCQ}

If we use the proposed algorithm:

$$\begin{aligned}
 \mathcal{A}_0 &= \{a : \leq 1r.(D \sqcap E), (a, b) : r, b : C \sqcap D, (a, c) : r, c : D \sqcap E, c : \neg C\} \\
 \mathcal{A}_1 &= \{a : \leq 1r.(D \sqcap E), (a, b) : r, b : C \sqcap D, (a, c) : r, c : D \sqcap E, c : \neg C\} \\
 &\quad \cup \{b : C, b : D\} \quad (\sqcap\text{-rule}) \\
 \mathcal{A}_2 &= \{a : \leq 1r.(D \sqcap E), (a, b) : r, b : C \sqcap D, (a, c) : r, c : D \sqcap E, c : \neg C, b : C, b : D\} \\
 &\quad \cup \{c : D, c : E\} \quad (\sqcap\text{-rule}) \\
 \mathcal{A}_3 &= \{a : \leq 1r.(D \sqcap E), (a, b) : r, b : C \sqcap D, (a, c) : r, c : D \sqcap E, c : \neg C, b : C, b : D, \\
 &\quad c : D, b : E\} \cup \{b : E\} \quad (\sqsubseteq\text{-rule})
 \end{aligned}$$

There is no rules can be applied and \mathcal{A}_3 is clash-free, the algorithm will return "consistent".

But \mathcal{K} is not consistent. If there is a model I w.r.t. \mathcal{K} . According to $b : C \sqcap D$ and $C \sqsubseteq E$ we can know that $b : D \sqcap E$, we can also know that $c : D \sqcap E$, but $I \models_{\mathcal{K}} a : \leq 1r.(D \sqcap E)$ which means there is at most 1 element in $(D \sqcap E)^I$ so it must be $b^I = c^I$. But $c : \neg C$ and $b \in C$, so there is a clash.

Therefore, the knowledge base isn't consistent but the proposed algorithm can't detect this.

Question 9. A Complex in ALC Extensions

(1) Let's prove a simple property first.

For any model \mathcal{I} w.r.t. \mathcal{T} :

$$\begin{aligned}
 C^{\mathcal{I}} \subseteq D^{\mathcal{I}} &\Leftrightarrow \forall a^{\mathcal{I}} \in \Delta^{\mathcal{I}}, a^{\mathcal{I}} \in C^{\mathcal{I}} \text{ implies } a^{\mathcal{I}} \in D^{\mathcal{I}} \\
 &\Leftrightarrow \forall a^{\mathcal{I}} \in \Delta^{\mathcal{I}}, a^{\mathcal{I}} \notin C^{\mathcal{I}} \text{ or } a^{\mathcal{I}} \in D^{\mathcal{I}} \\
 &\Leftrightarrow \forall a^{\mathcal{I}} \in \Delta^{\mathcal{I}}, a^{\mathcal{I}} \in (\neg C)^{\mathcal{I}} \text{ or } a^{\mathcal{I}} \in D^{\mathcal{I}} \\
 &\Leftrightarrow \forall a^{\mathcal{I}} \in \Delta^{\mathcal{I}}, a^{\mathcal{I}} \in (\neg C \sqcup D)^{\mathcal{I}} \\
 &\Leftrightarrow \Delta^{\mathcal{I}} \subseteq (\neg C \sqcup D)^{\mathcal{I}}
 \end{aligned}$$

So if I is a model of \mathcal{T} , then:

$$\begin{aligned}
 &\text{for each GCI } C \sqsubseteq D \in \mathcal{T}, C^I \subseteq D^I \\
 &\Leftrightarrow \text{for each GCI } C \sqsubseteq D \in \mathcal{T}, \Delta^I \subseteq (\neg C \sqcup D)^I \\
 &\Leftrightarrow \Delta^I \subseteq \bigcap_{C \sqsubseteq D \in \mathcal{T}} (\neg C \sqcup D)^I \\
 &\Leftrightarrow I \text{ satisfy } \top \sqsubseteq \bigcap_{C \sqsubseteq D \in \mathcal{T}} \neg C \sqcup D \\
 &\Leftrightarrow I \text{ is a model of } \mathcal{T}'
 \end{aligned}$$

Therefore, \mathcal{T} and \mathcal{T}' have the same models.

(2) \implies :

If \mathcal{K} is consistent, then there exists a model I for \mathcal{K} such that $I \models \mathcal{T}$, $I \models \mathcal{A}$, and $I \models \mathcal{R}$. So what remain to us to prove is \mathcal{I} satisfies \mathcal{T}^+ , i.e. $(\forall r.C)^{\mathcal{I}} \subseteq (\forall r.\forall r.C)^{\mathcal{I}}$.

For each element a^I such that $a^I \in (\forall r.C)^{\mathcal{I}}$, we consider two cases:

1. If there is not b^I such that $(a^I, b^I) \in r^I$, then $b^I \in (\forall r, \forall r.C)^{\mathcal{I}}$.
2. If there exists some $b^I \in \Delta^I$ such that $(a^I, b^I) \in r^I$, then $b^I \in C^I$. If $b^I \notin (\forall r.C)^{\mathcal{I}}$, then there must exist $c^I \in (\neg C)^{\mathcal{I}}$ such that $(b^I, c^I) \in r^I$. However, transitivity of r implies $(a^I, c^I) \in r^I$, and hence $c^I \in C^I$, which is a contradiction. Therefore, $b^I \in (\forall r.C)^{\mathcal{I}}$. Therefore, we have $a^I \in (\forall r.\forall r.C)^{\mathcal{I}}$.

Therefore, $(\forall r.C)^{\mathcal{I}} \subseteq (\forall r.\forall r.C)^{\mathcal{I}}$, which means $\mathcal{I} \models \forall r.C \sqsubseteq \forall r.\forall r.C$, \mathcal{I} is a model of \mathcal{T}^+ .

So we have proved that \mathcal{K}^+ is consistent.

\Longleftarrow :

If \mathcal{K}^+ is consistent, then there exists a model \mathcal{I} for \mathcal{K}^+ . Because $\mathcal{T} \subseteq \mathcal{T}^+$, so $\mathcal{I} \models (\mathcal{T}, \mathcal{A})$. So what remain to us to prove is \mathcal{I} satisfies \mathcal{R} .

Considering the form: $\forall r.C \sqsubseteq \forall r.\forall r.C \in \mathcal{T}^+$ for all $\text{tran}(s) \in \mathcal{R}$.

For all $(x, z) \in r^{\mathcal{I}}$, there are $y \in \Delta^{\mathcal{I}}$ such that $(x, y) \in r^{\mathcal{I}}$ and $(y, z) \in r^{\mathcal{I}}$, which means r is transitive w.r.t. \mathcal{I} , so \mathcal{I} is a model of \mathcal{R} .

So we have proved that \mathcal{K} is consistent.

Question 10 (with 1 bonus mark). Pushdown automata

(1) Let $A = \{0^k 1^k 2^k \mid k \geq 0\}$.

We firstly show that A can be recognized by 2-PDAs.

The procedure is as follow:

- read '0' from input, and push it to stack 1 until read '1'. If read symbol that is not '0' or '1', reject.
- read '1' from input, and push it to stack 2 until read '2'. If read symbol that is not '1' or '2', reject.
- read '2' from input, pop one element from both stack 1 and stack 2. If read symbol that is not '2', reject.
- When read the whole string, it will accept it if both stack 1 and stack 2 are empty, otherwise reject.

Then we show that A could not be recognized by 1-PDAs.

We prove that A is not a CFL by contradiction.

If A is a CFL, then it satisfies the pumping lemma of CFL, which is as follow:

- (1) for each $i \geq 0$, $uv^i xy^i z \in A$.
- (2) $vy \neq \varepsilon$.
- (3) $|vxy| \leq p$.

$s = 0^p 1^p 2^p$, condition (3) implies that there are at most two kinds of symbols occurs at vxy .

Therefore, there are 0s, 1s and 2s with unequal length in $uv^2 xy^2 z$, which means $uv^2 xy^2 z \notin B$.

So B is not a CFL, it could not be recognized by 1-PDAs. Through this example, we can see that 2-PDAs are more powerful than 1-PDAs.

- (2) We can use 2 stacks to simulate 3 stacks.

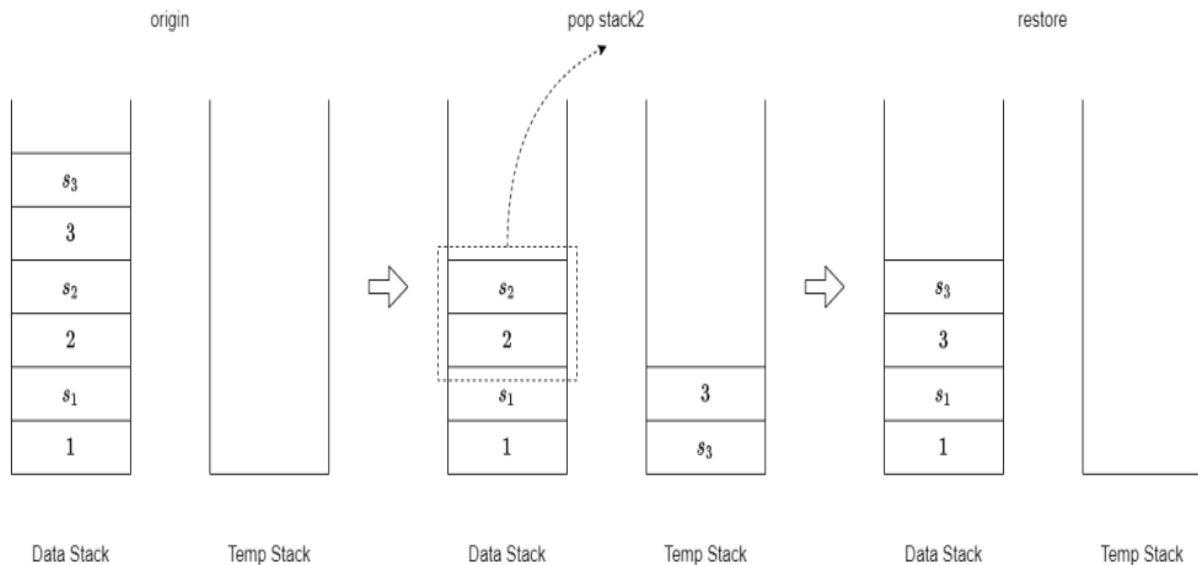
Let a 3-PDA $A = \langle Q, \Sigma, \Gamma, \delta, q_0, F \rangle$.

Let a 2-PDA $A = \langle Q, \Sigma, \Gamma', \delta, q_0, F \rangle$, $\Gamma' = \Gamma \cup \{s_1, s_2, s_3\}$, where s_1 , s_2 and s_3 are label elements to identify in which stack elements of data stack are.

If we need to push a element, we firstly push the element into stack 1 (we also call it data stack), then we push a label element into data stack.

If we need to pop a element with specific label, such as label s_2 (which means it belongs to stack 2 in 3-PDAs), we firstly pop all elements whose label is not s_2 from top of data stack, and push them into stack 2 (we also call it temp stack). Once we read a label element s_2 and then pop its data element, finally restore all elements in temp stack into data stack.

See the following picture:



So 3-PDAs are not more powerful than 2-PDAs.

Question 11 (with 1 bonus mark). Turing machine

1. Scan the input string from left to right to see whether the read symbol is a

member of $\{0, 1\}^*$. If there is a symbol which is not 0s or 1s, reject, otherwise go to step 2.

2. Move the head to the left-hand end, and move to right until read a 0s, write 's' and go to step 3. If there is no 0s, go to step 4.
3. Move the head to the left-hand end, and move to right until read a 1s, write 's' and go to step 2. If there is no 1s, reject.
4. Move the head to the left-hand end, and scan from left to right, if there is a symbol other than 's', reject, otherwise accept.

Question 12 (with 1 bonus mark). Decidability

A is decidable.

There are only 2 possibilities for set A : either $\{0\}$ or $\{1\}$, which are both finite sets and hence decidable.

We can define a TM to decides it: Match the input string with the only string s in language. Accept if they match, and reject otherwise.

Therefore, even though we don't know whether there will be reasonably-priced yummys in the canteens of NJU someday, the language A is decidable.