

Abstract Syntax Networks for Code Generation and Semantic Parsing

Berkeley



Maxim Rabinovich*, Mitchell Stern*, Dan Klein



Code Generation





Code Generation



Input

```
class DireWolfAlpha(MinionCard):
    def __init__(self):
        super().__init__(
            "Dire Wolf Alpha", 2,
            CHARACTER_CLASS.ALL,
            CARD_RARITY.COMMON,
            minion_type=MINION_TYPE.BEAST)

    def create_minion(self, player):
        return Minion(2, 2, auras=[
            Aura(ChangeAttack(1),
                MinionSelector(Adjacent()))
        ])
```

Output

[Ling et al. 2016]



Code Generation



Input

```
class DireWolfAlpha(MinionCard):
    def __init__(self):
        super().__init__(
            "Dire Wolf Alpha", 2,
            CHARACTER_CLASS.ALL,
            CARD_RARITY.COMMON,
            minion_type=MINION_TYPE.BEAST)

    def create_minion(self, player):
        return Minion(2, 2, auras=[
            Aura(ChangeAttack(1),
                MinionSelector(Adjacent()))
        ])
```

Output

[Ling et al. 2016]



Code Generation



Input

```
class DireWolfAlpha(MinionCard):
    def __init__(self):
        super().__init__(
            "Dire Wolf Alpha", 2,
            CHARACTER_CLASS.ALL,
            CARD_RARITY.COMMON,
            minion_type=MINION_TYPE.BEAST)

    def create_minion(self, player):
        return Minion(2, 2, auras=[
            Aura(ChangeAttack(1),
                MinionSelector(Adjacent()))
        ])
```

Output

[Ling et al. 2016]



Code Generation



Input

```
class DireWolfAlpha(MinionCard):
    def __init__(self):
        super().__init__(
            "Dire Wolf Alpha", 2,
            CHARACTER_CLASS.ALL,
            CARD_RARITY.COMMON,
            minion_type=MINION_TYPE.BEAST)

    def create_minion(self, player):
        return Minion(2, 2, auras=[
            Aura(ChangeAttack(1),
                MinionSelector(Adjacent()))
        ])
```

Output

[Ling et al. 2016]



Code Generation



Input

```
class DireWolfAlpha(MinionCard):
    def __init__(self):
        super().__init__(
            "Dire Wolf Alpha", 2,
            CHARACTER_CLASS.ALL,
            CARD_RARITY.COMMON,
            minion_type=MINION_TYPE.BEAST)

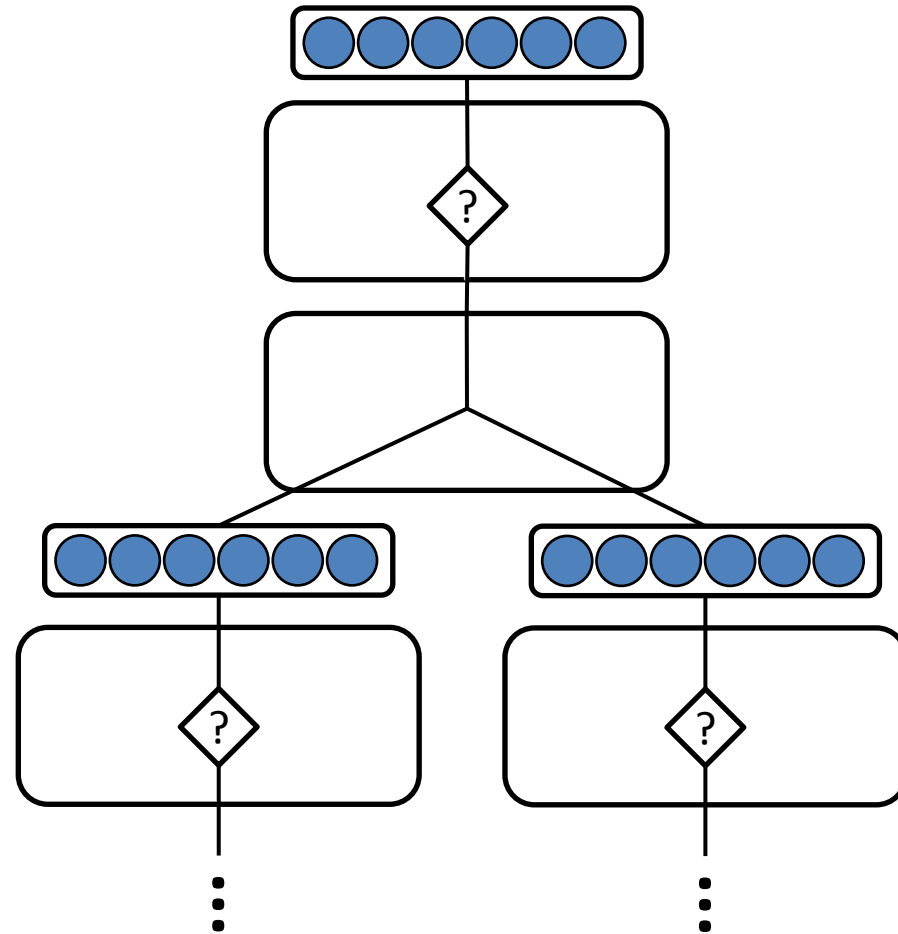
    def create_minion(self, player):
        return Minion(2, 2, auras=[
            Aura(ChangeAttack(1),
                MinionSelector(Adjacent()))
        ])
```

Output

[Ling et al. 2016]



The Main Idea





Card Representation

name	Dire Wolf Alpha
cost	2
type	Minion
rarity	Common
race	Beast
class	Neutral
health	2
attack	2
description	Adjacent minions have +1 Attack.

Input

```
class DireWolfAlpha(MinionCard):
    def __init__(self):
        super().__init__(
            "Dire Wolf Alpha", 2,
            CHARACTER_CLASS.ALL,
            CARD_RARITY.COMMON,
            minion_type=MINION_TYPE.BEAST)

    def create_minion(self, player):
        return Minion(2, 2, auras=[
            Aura(ChangeAttack(1),
                MinionSelector(Adjacent()))
        ])
```

Output

[Ling et al. 2016]



Code Representations

```
Aura (ChangeAttack (1), MinionSelector (Adjacent ()))
```



Code Representations

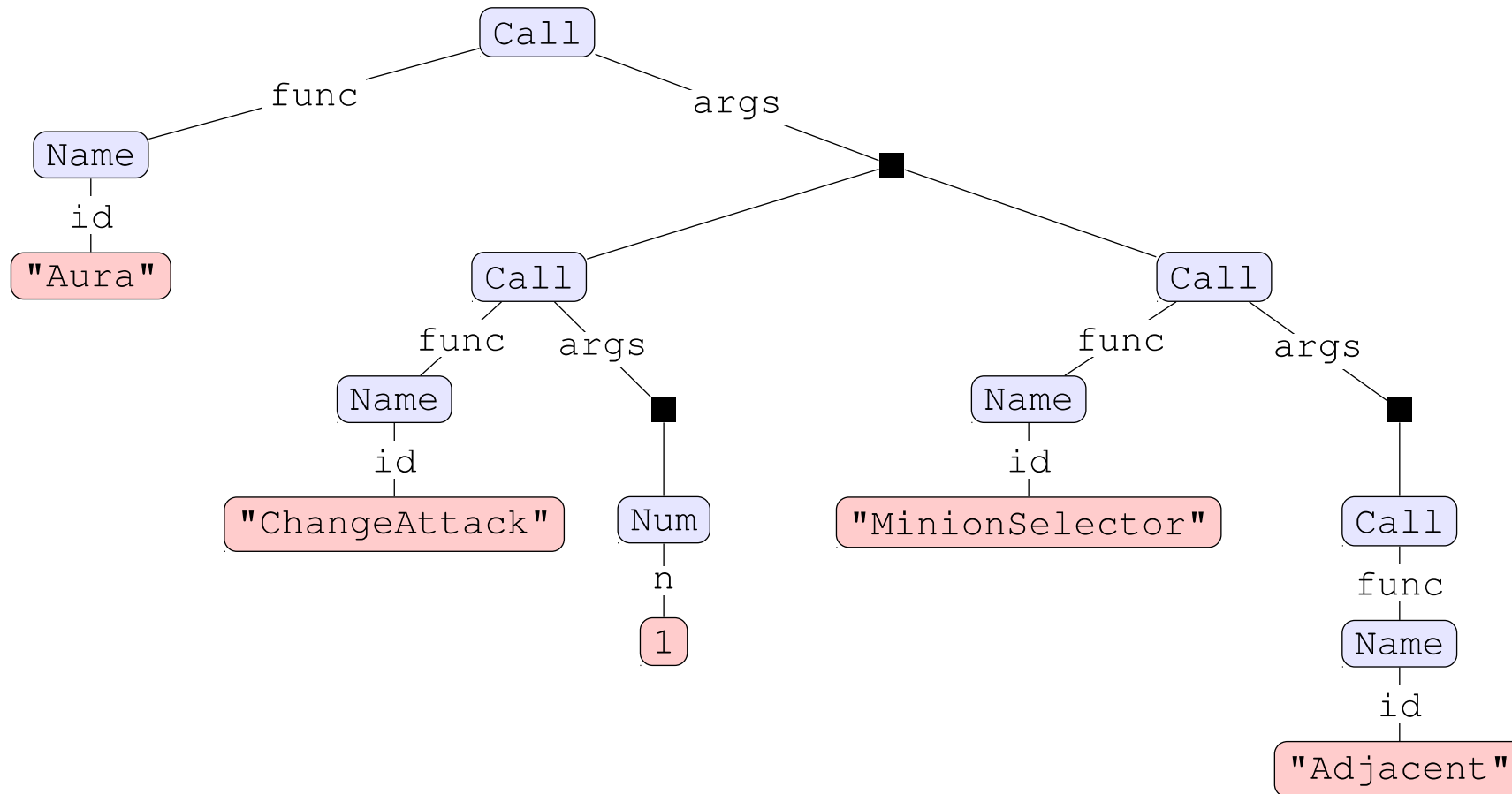
```
Aura(ChangeAttack(1), MinionSelector(Adjacent()))
```

```
Aura ( ChangeAttack ( 1 ) , MinionSelector ( Adjacent ( ) ) )
```



Code Representations

`Aura (ChangeAttack (1) , MinionSelector (Adjacent ()))`



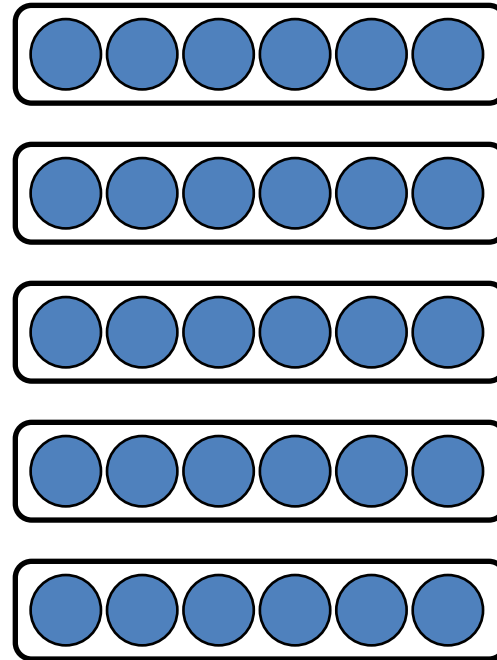
`Aura (ChangeAttack (1) , MinionSelector (Adjacent ()))`



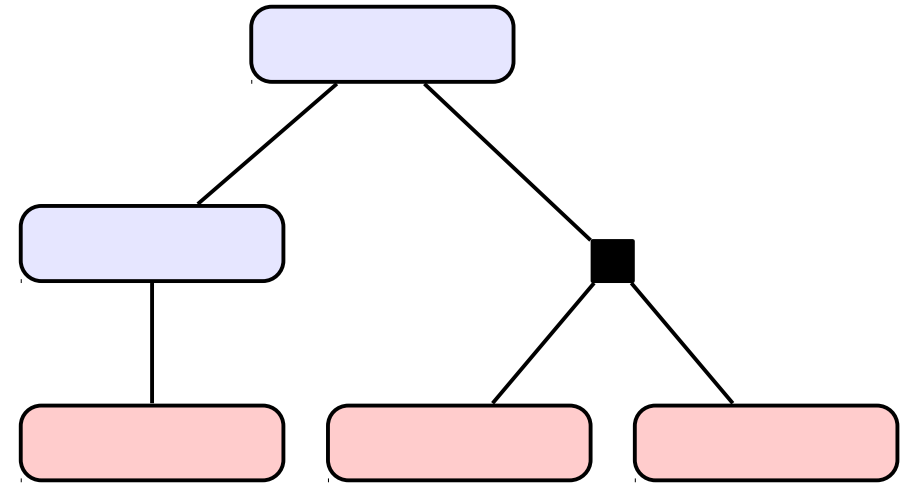
Model Overview



Input



Encoding

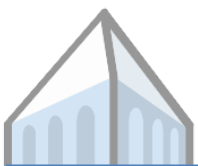


Output



Encoding





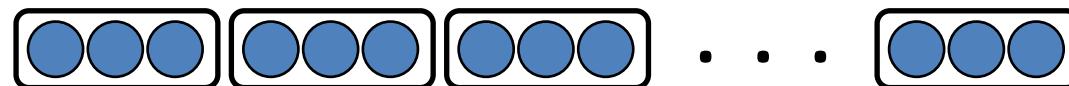
Encoding

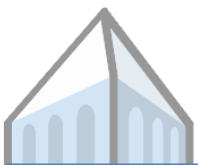
name	Dire Wolf Alpha
cost	2
type	Minion
rarity	Common
race	Beast
class	Neutral
health	2
attack	2
description	Adjacent minions have +1 Attack.



Encoding

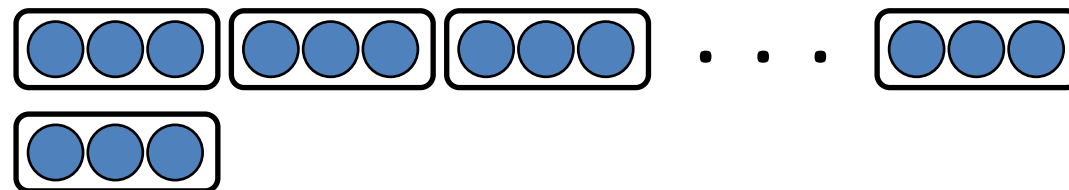
name	Dire Wolf Alpha
cost	2
type	Minion
rarity	Common
race	Beast
class	Neutral
health	2
attack	2
description	Adjacent minions have +1 Attack.

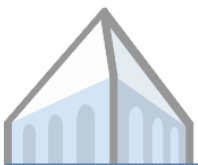




Encoding

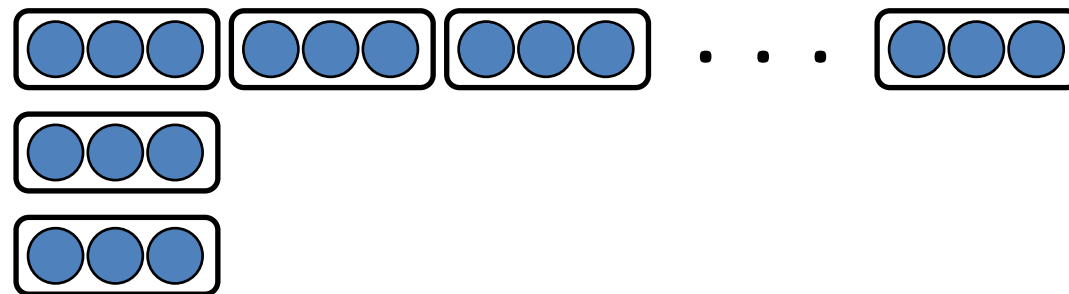
name	Dire Wolf Alpha
cost	2
type	Minion
rarity	Common
race	Beast
class	Neutral
health	2
attack	2
description	Adjacent minions have +1 Attack.





Encoding

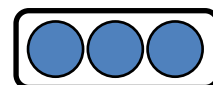
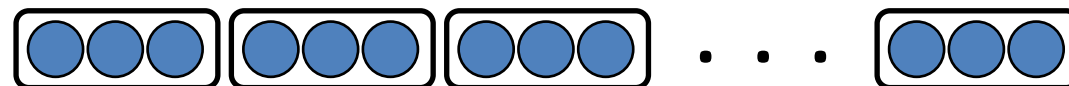
name	Dire Wolf Alpha
cost	2
type	Minion
rarity	Common
race	Beast
class	Neutral
health	2
attack	2
description	Adjacent minions have +1 Attack.

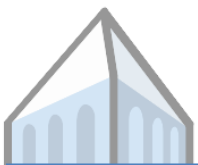





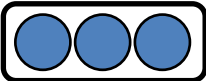
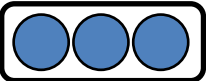



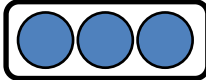
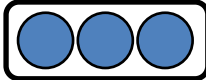



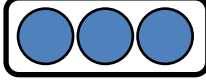
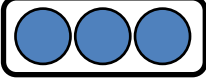
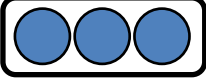
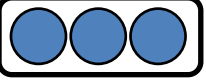
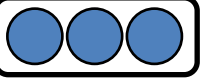
Encoding

name	Dire Wolf Alpha
cost	2
type	Minion
rarity	Common
race	Beast
class	Neutral
health	2
attack	2
description	Adjacent minions have +1 Attack.





Encoding

name	Dire Wolf Alpha	   . . . 
cost	2	
type	Minion	
rarity	Common	
race	Beast	
class	Neutral	
health	2	
attack	2	
description	Adjacent minions have +1 Attack.	    

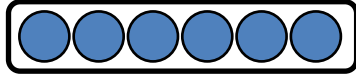


Decoder Architecture





Decoder Architecture



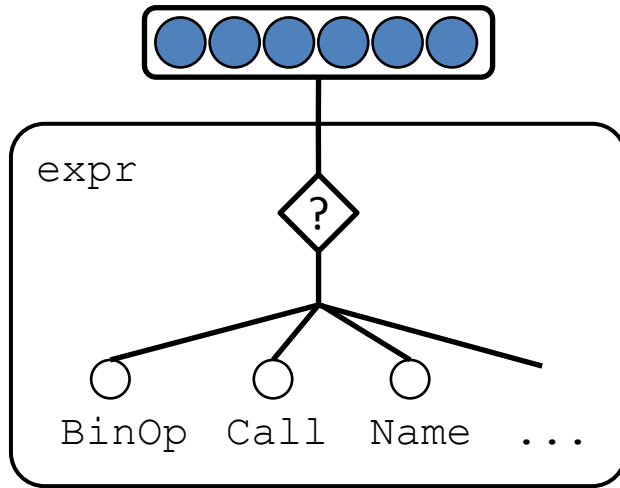


Decoder Architecture



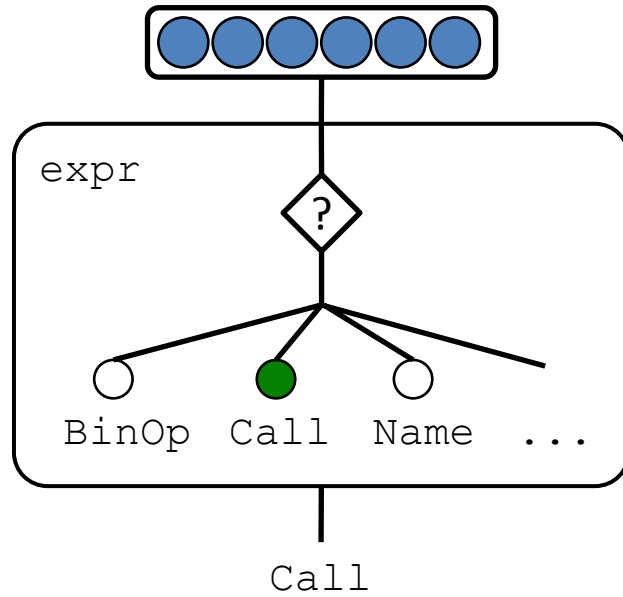


Decoder Architecture



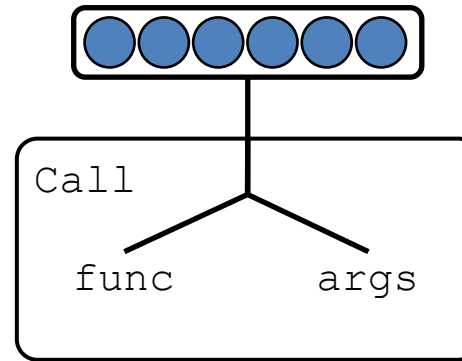
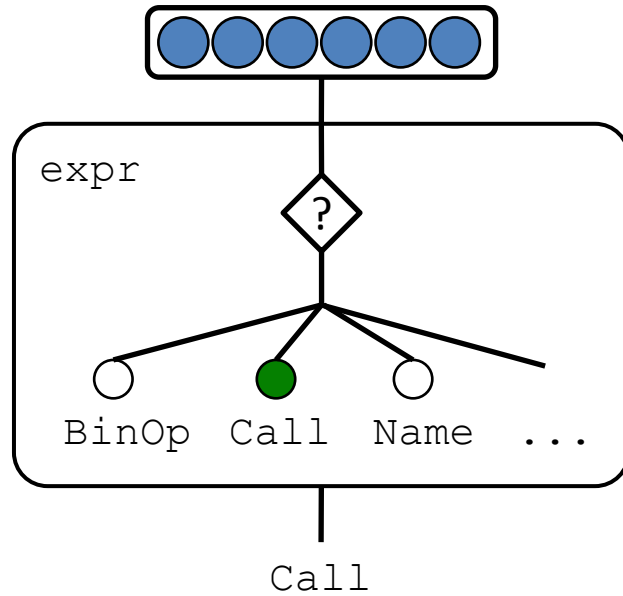


Decoder Architecture



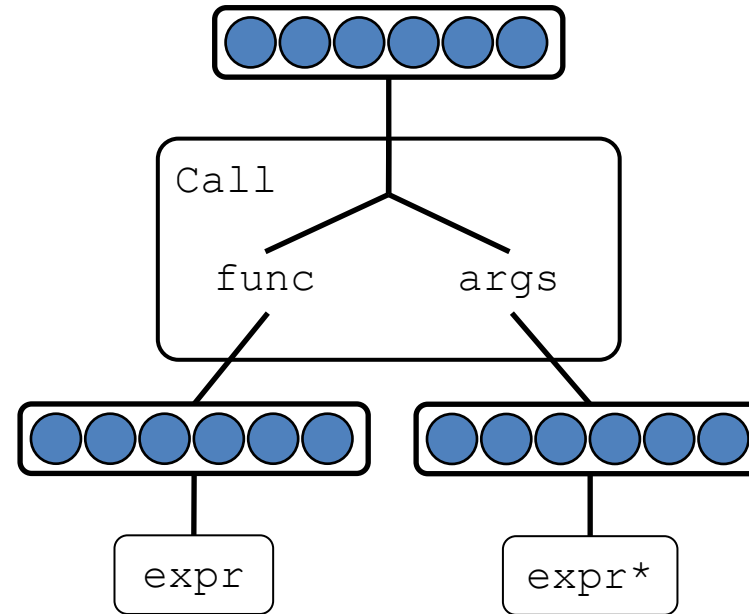
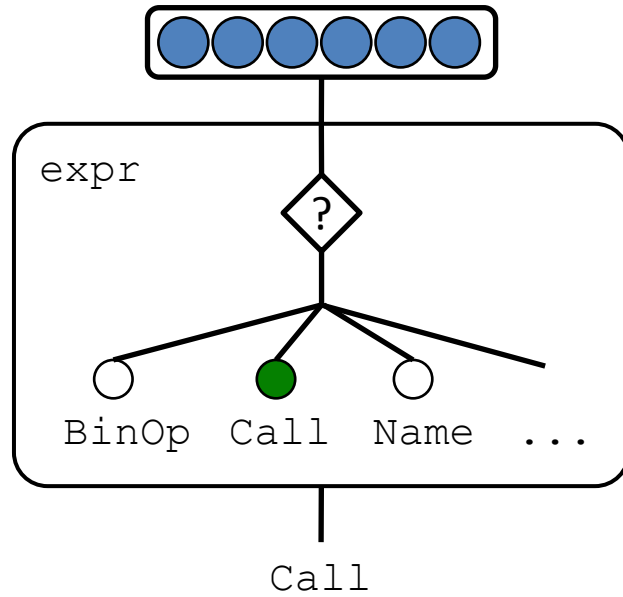


Decoder Architecture



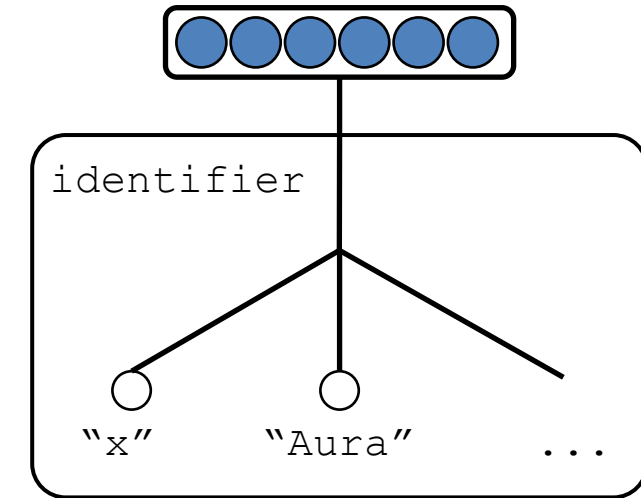
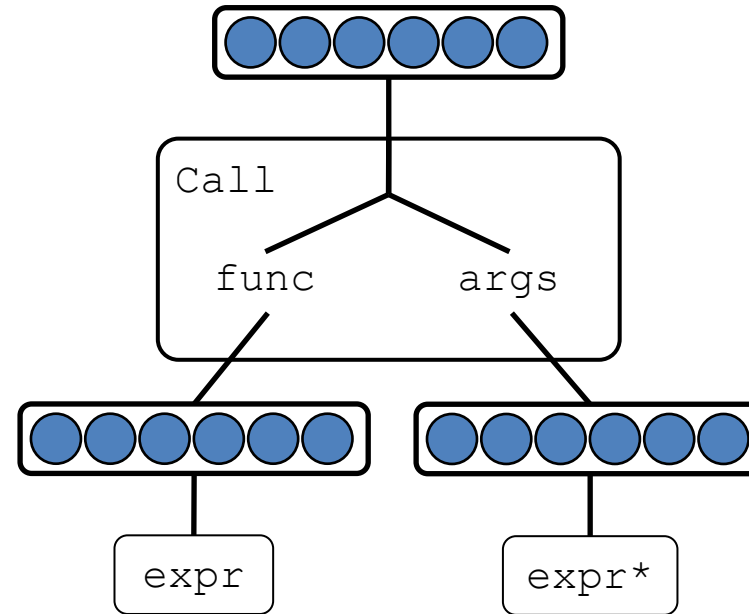
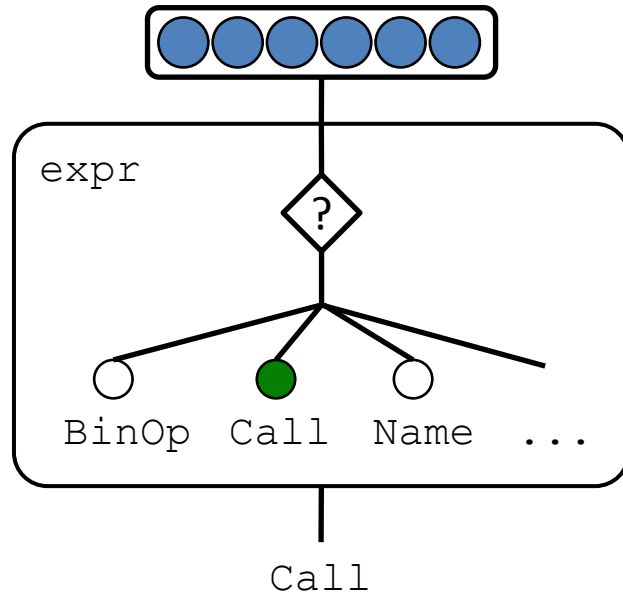


Decoder Architecture



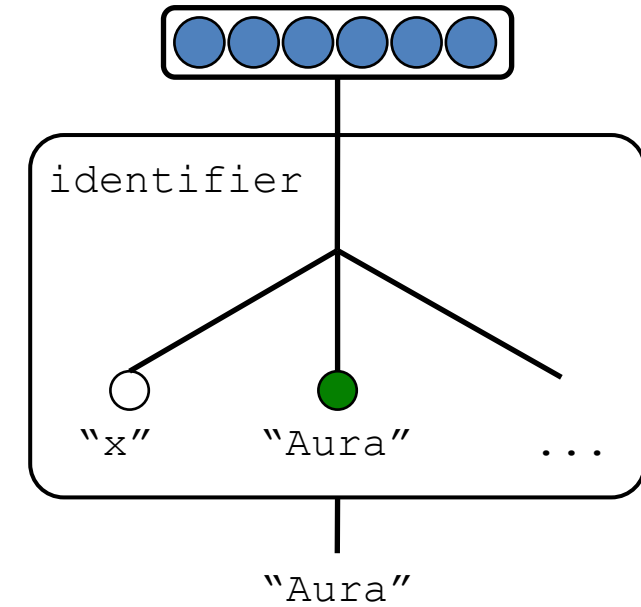
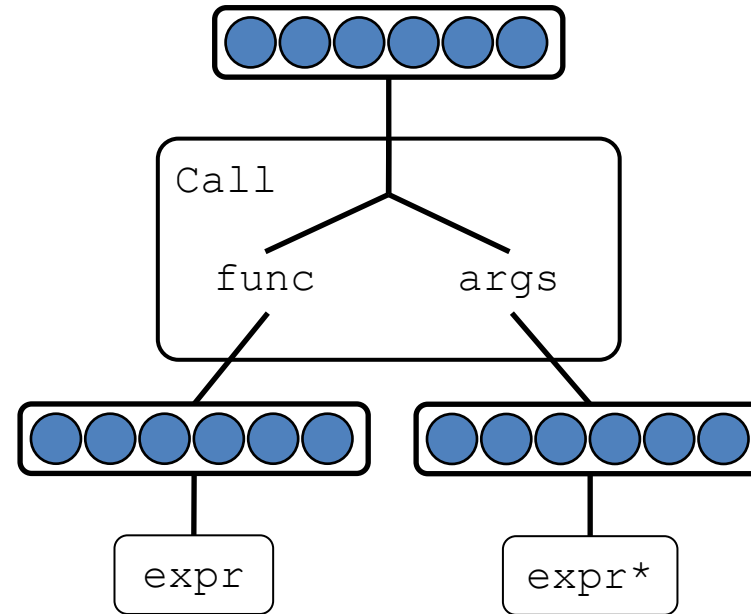
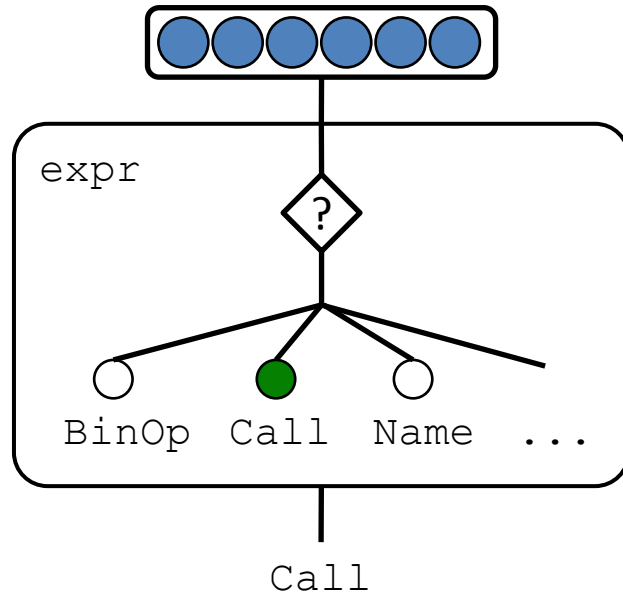


Decoder Architecture



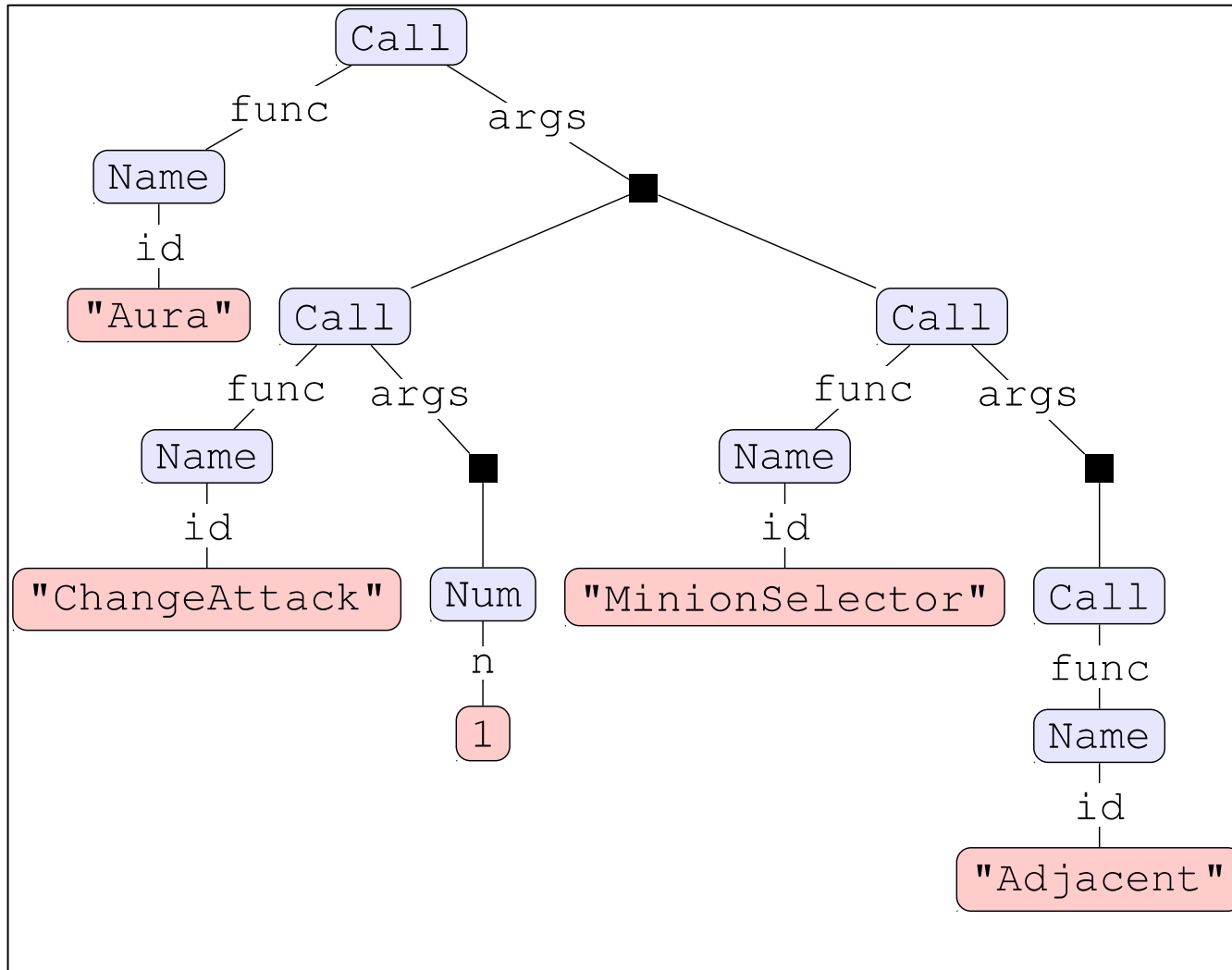


Decoder Architecture





Decoding Process





Decoding Process

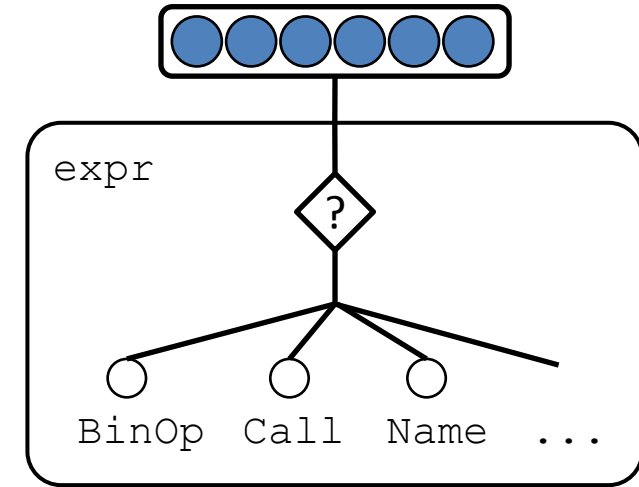
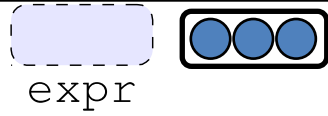


expr



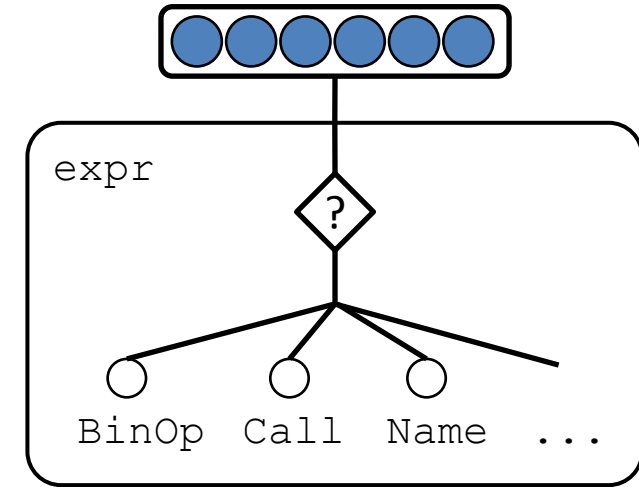
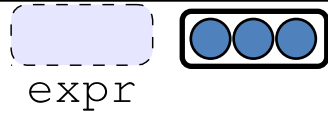


Decoding Process



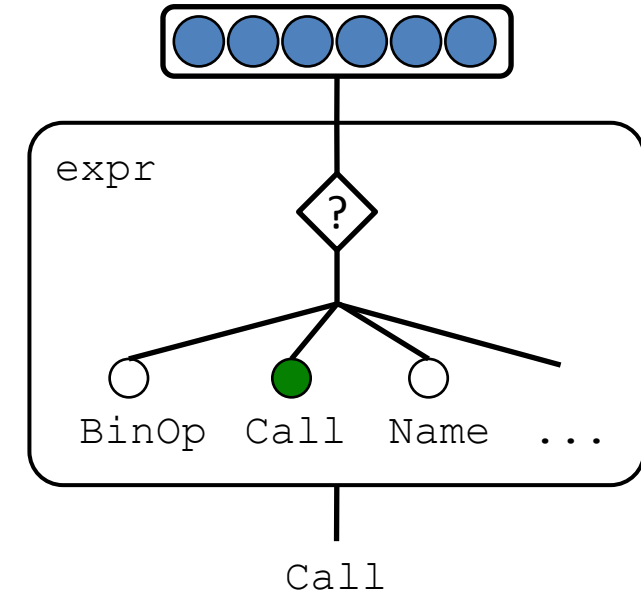
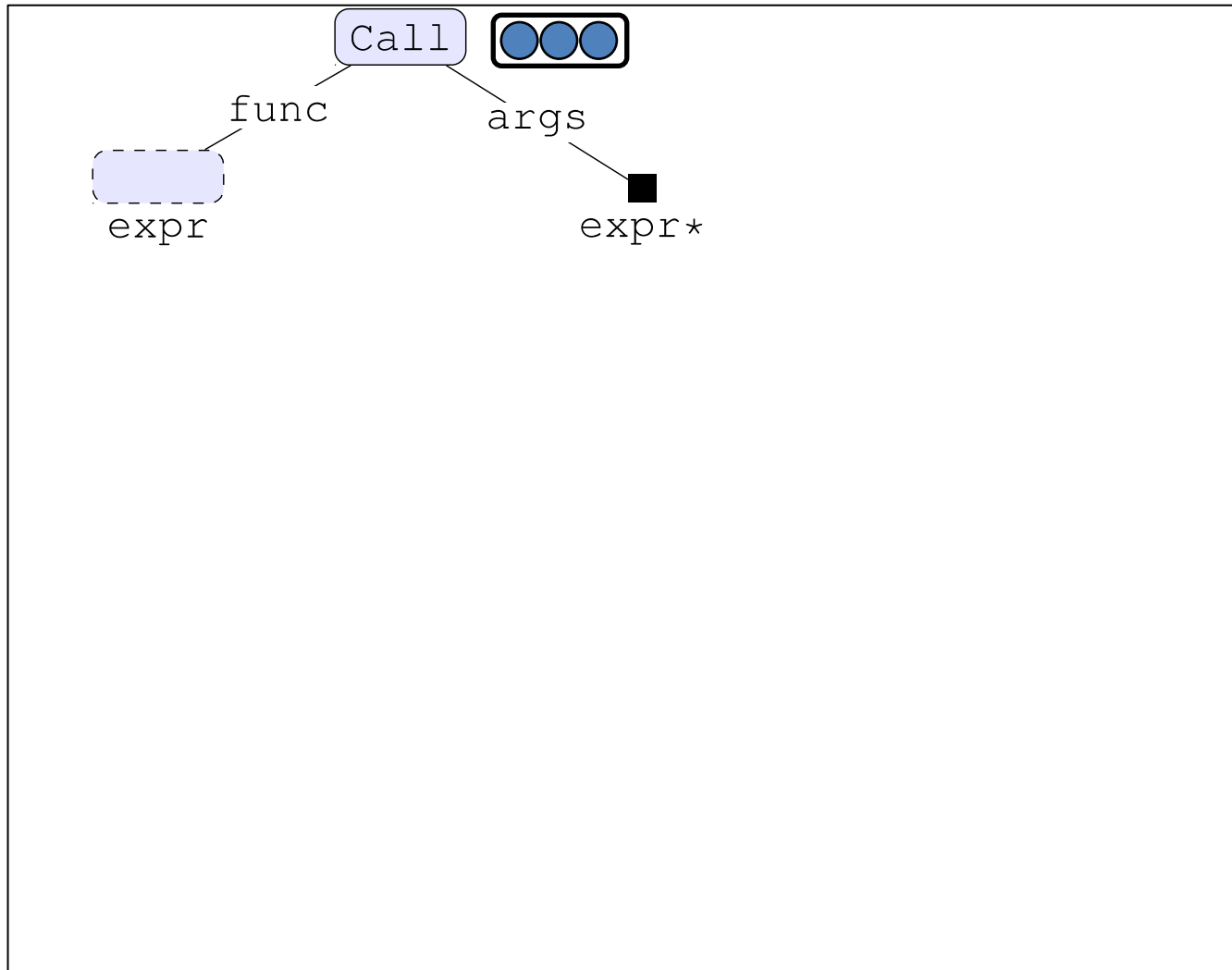


Decoding Process



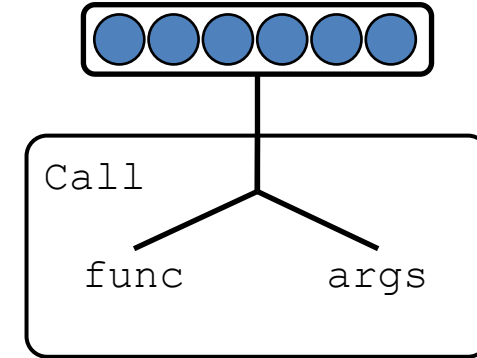
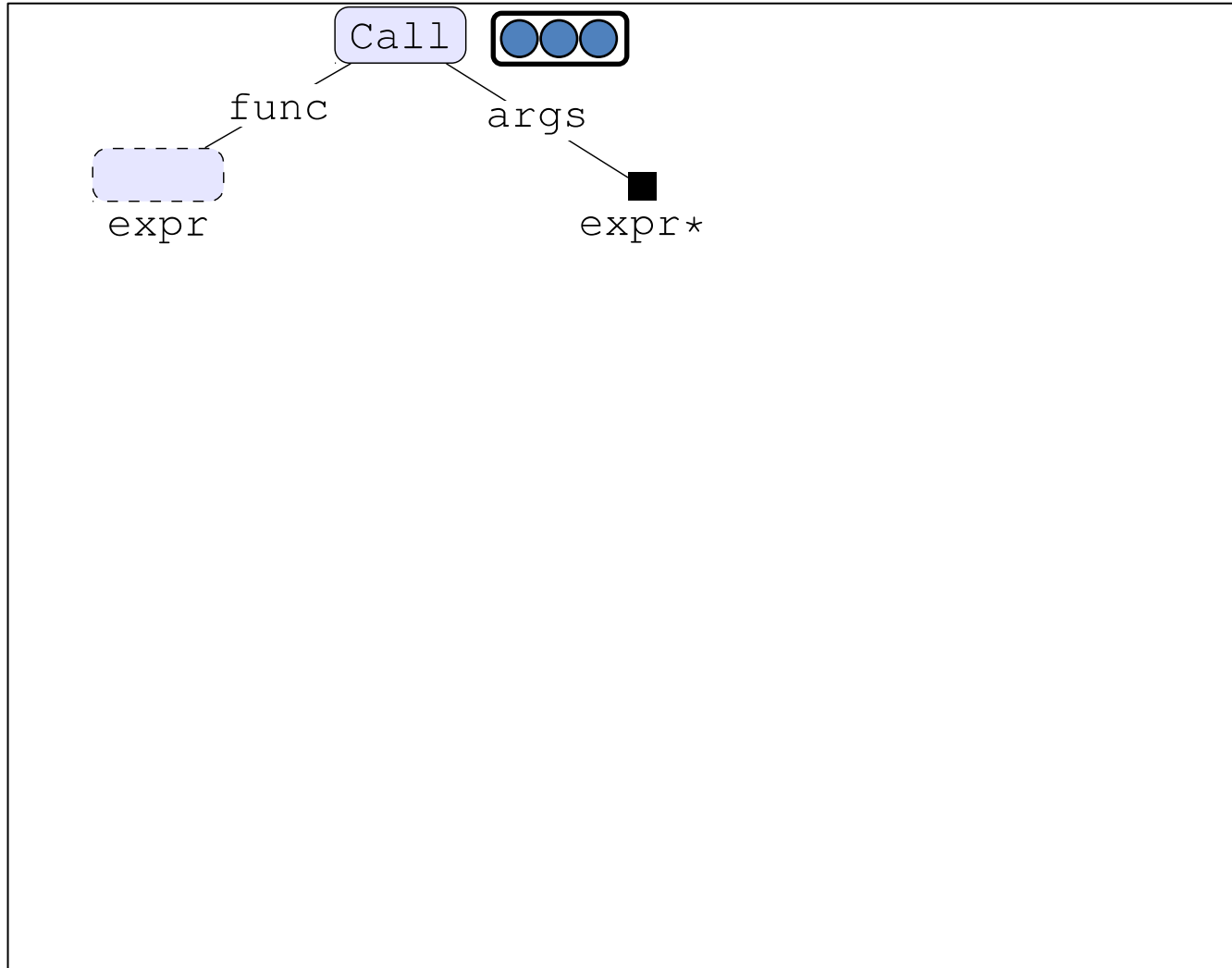


Decoding Process



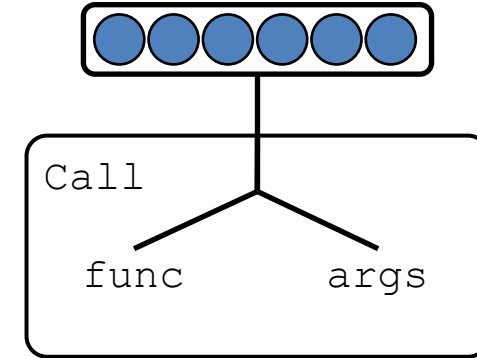
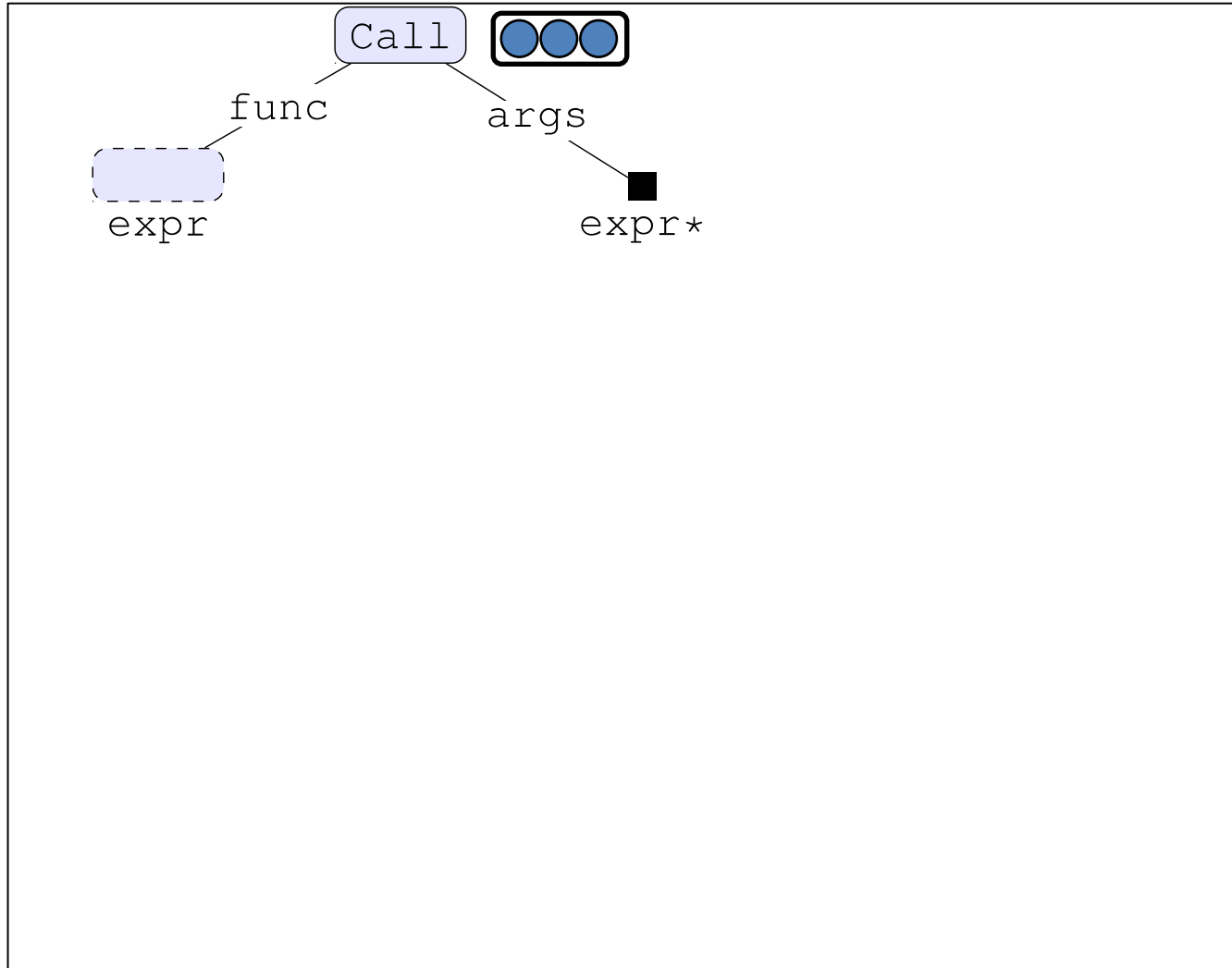


Decoding Process



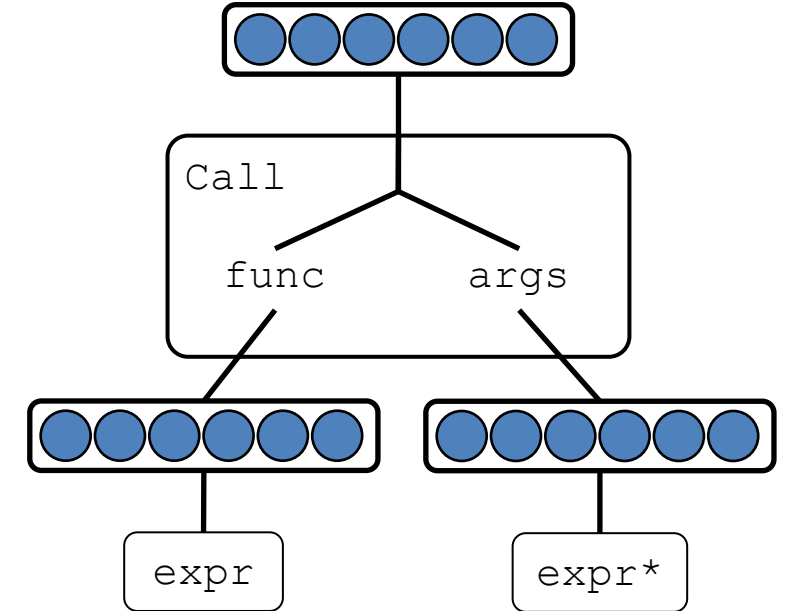
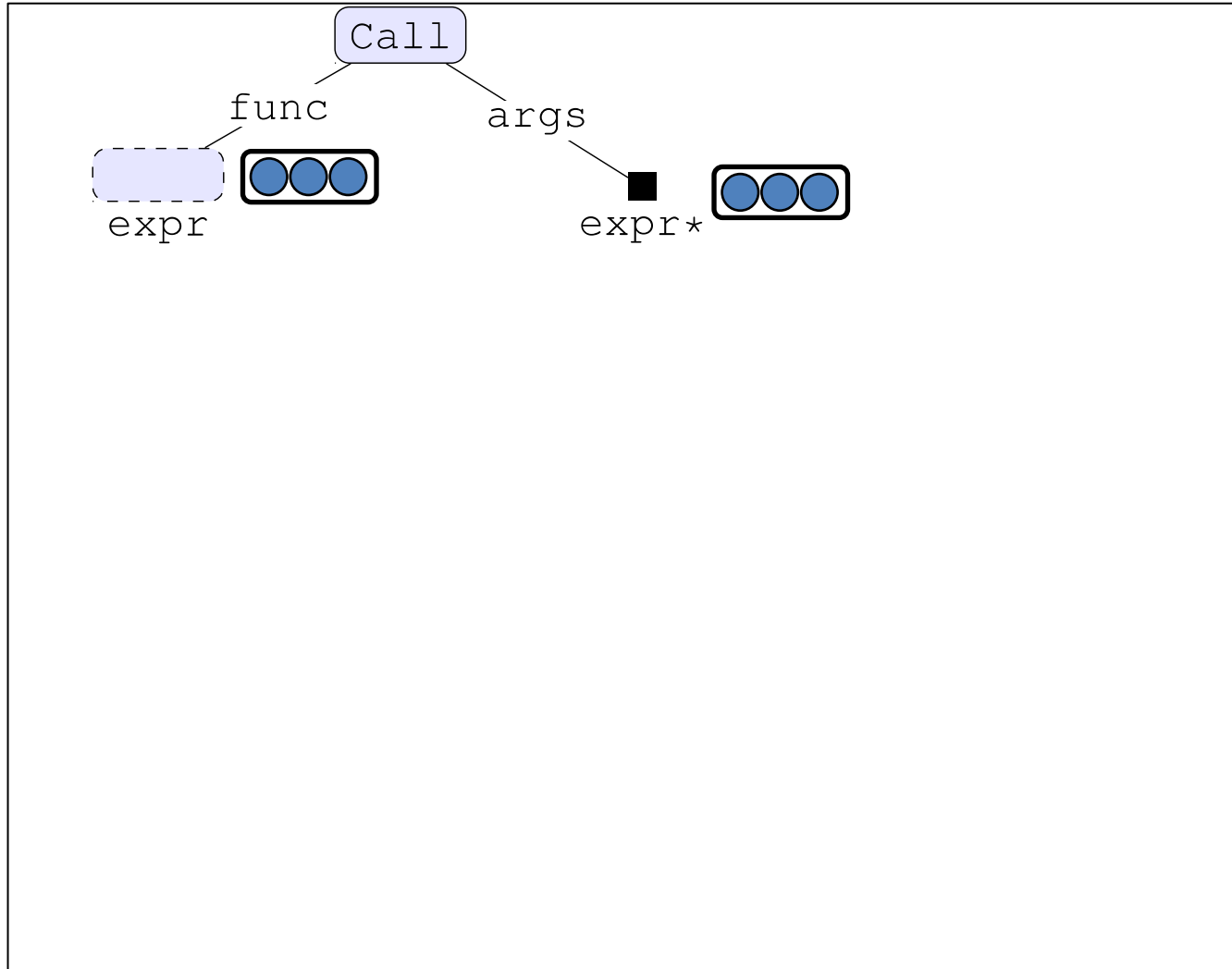


Decoding Process



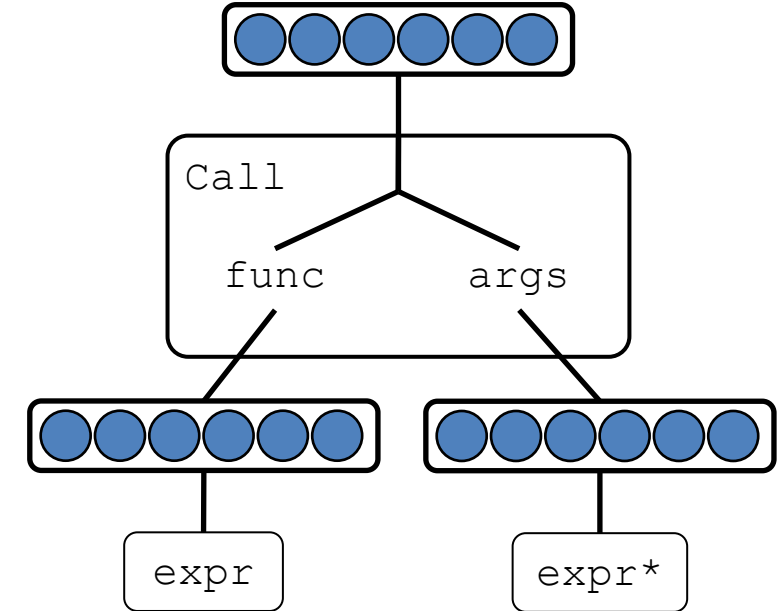
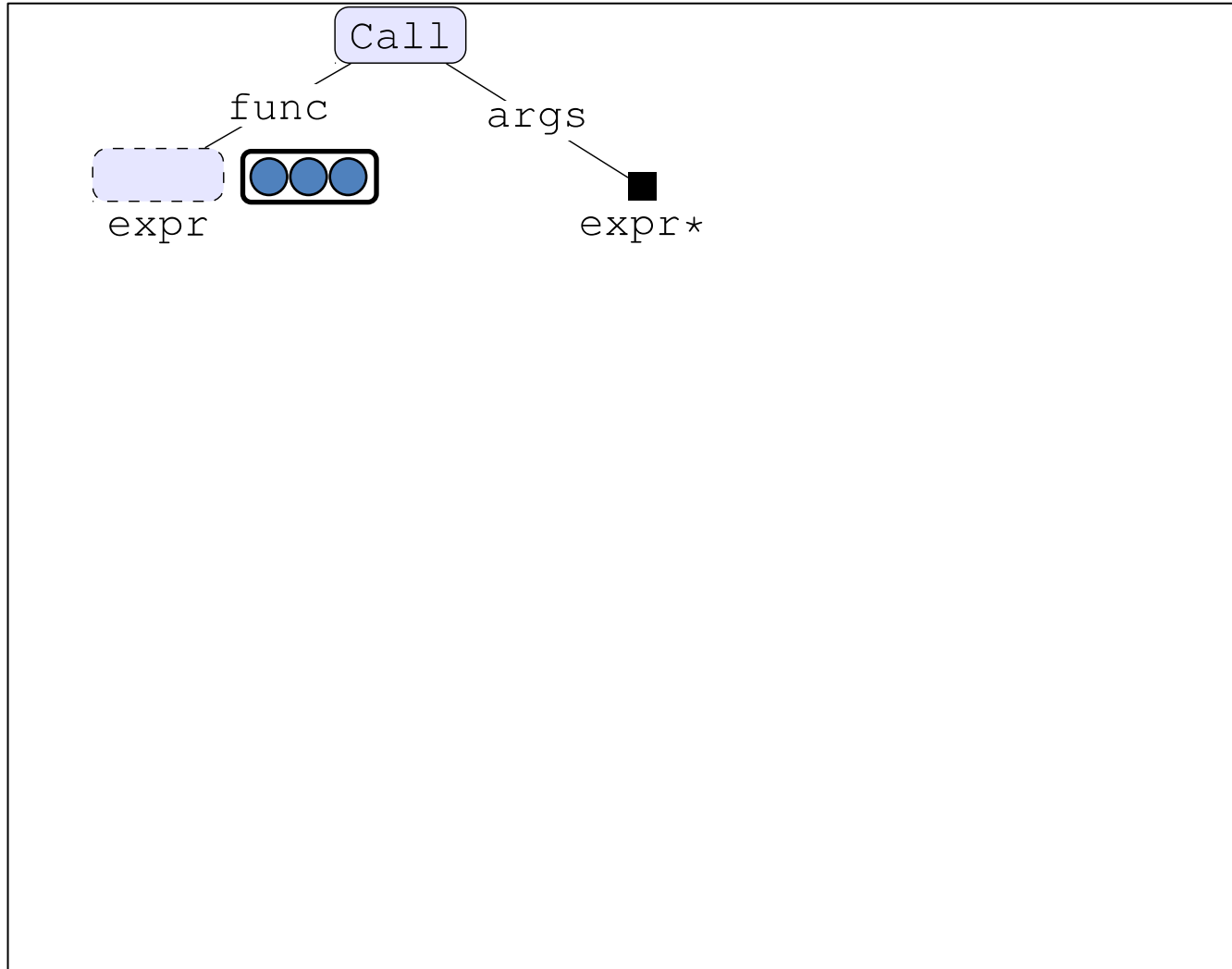


Decoding Process



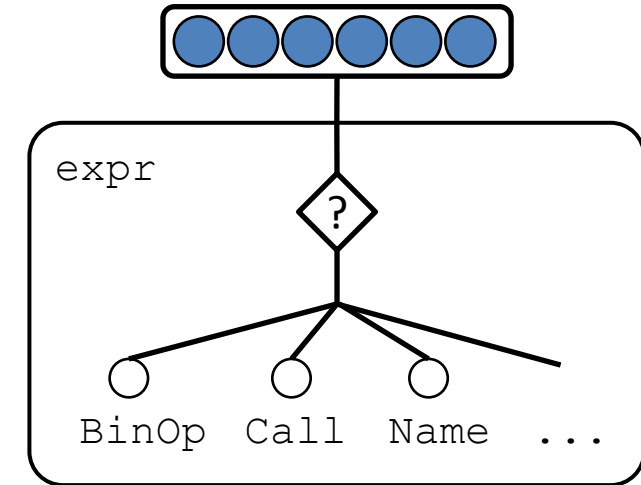
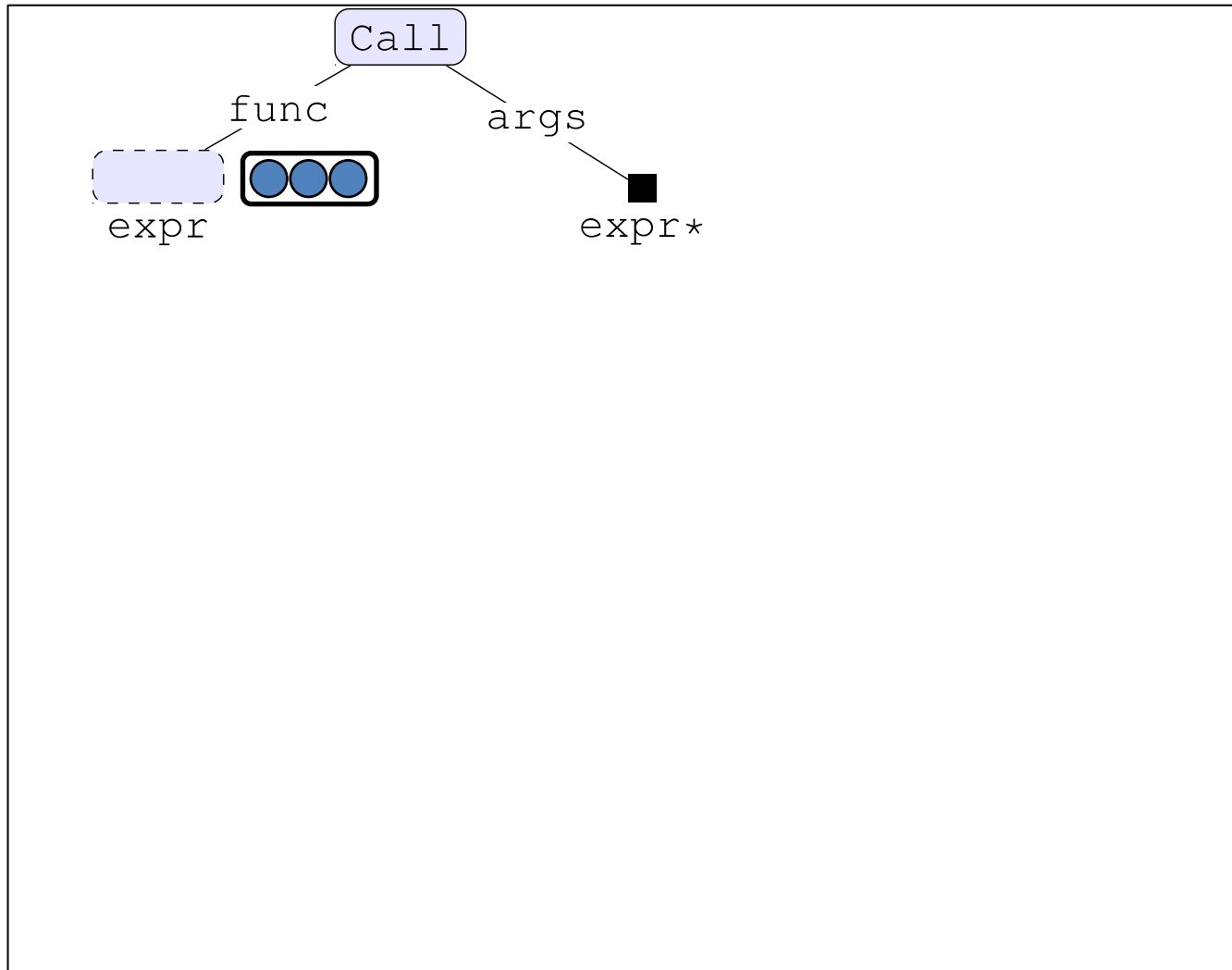


Decoding Process



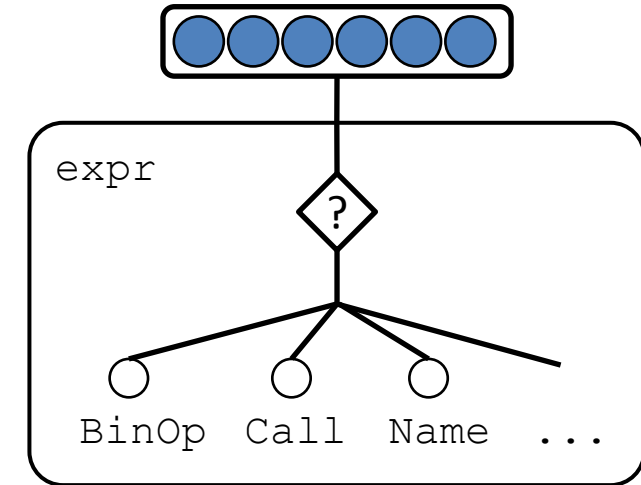
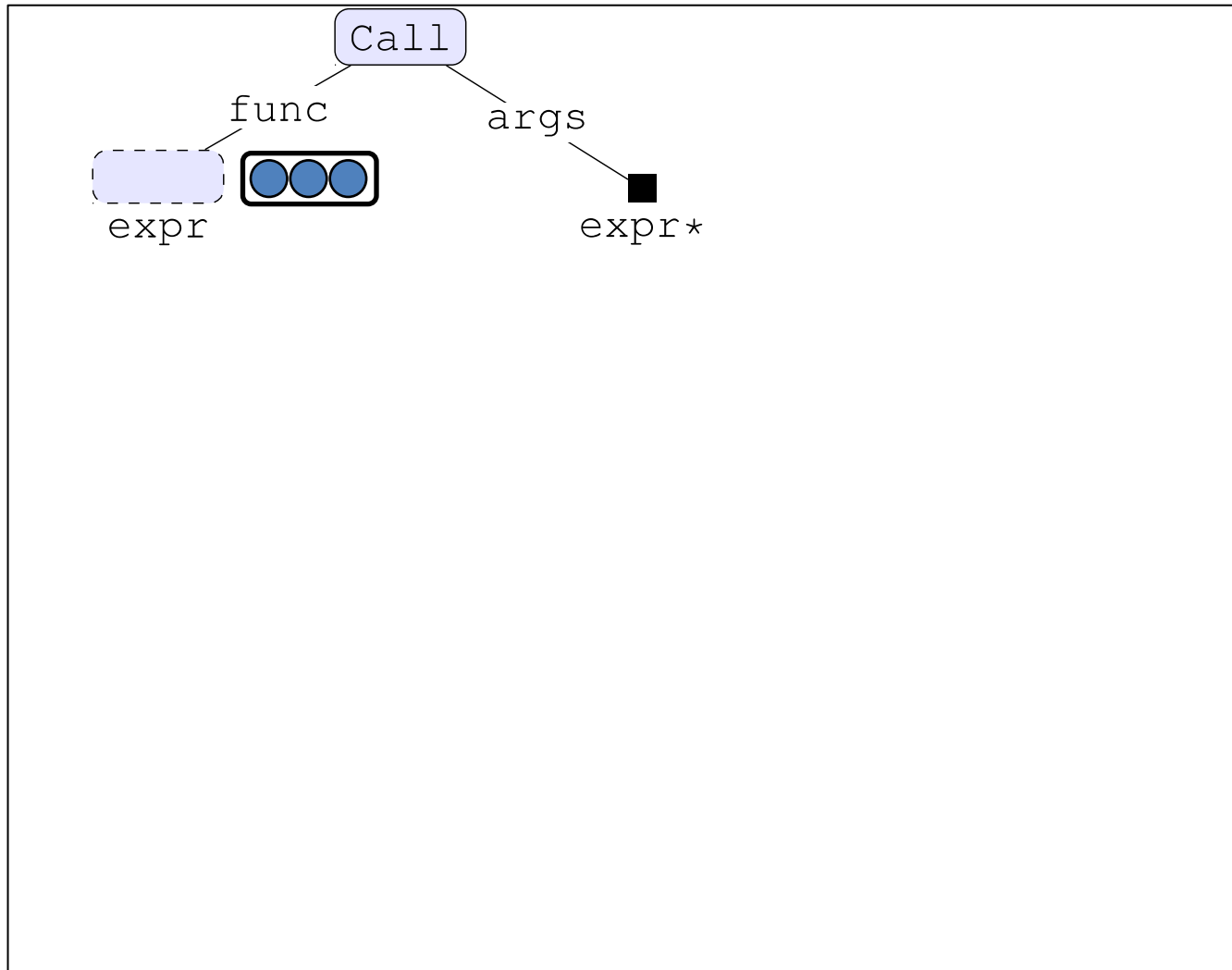


Decoding Process



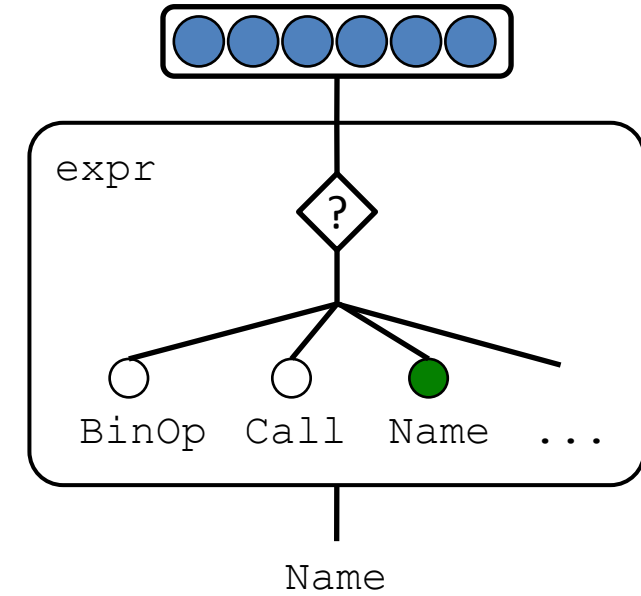
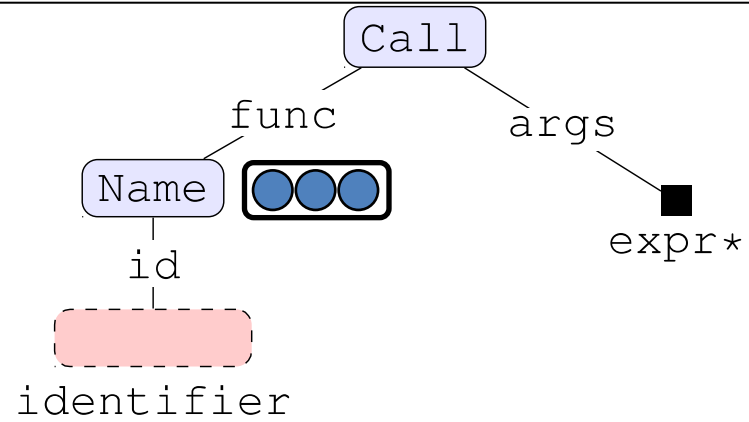


Decoding Process



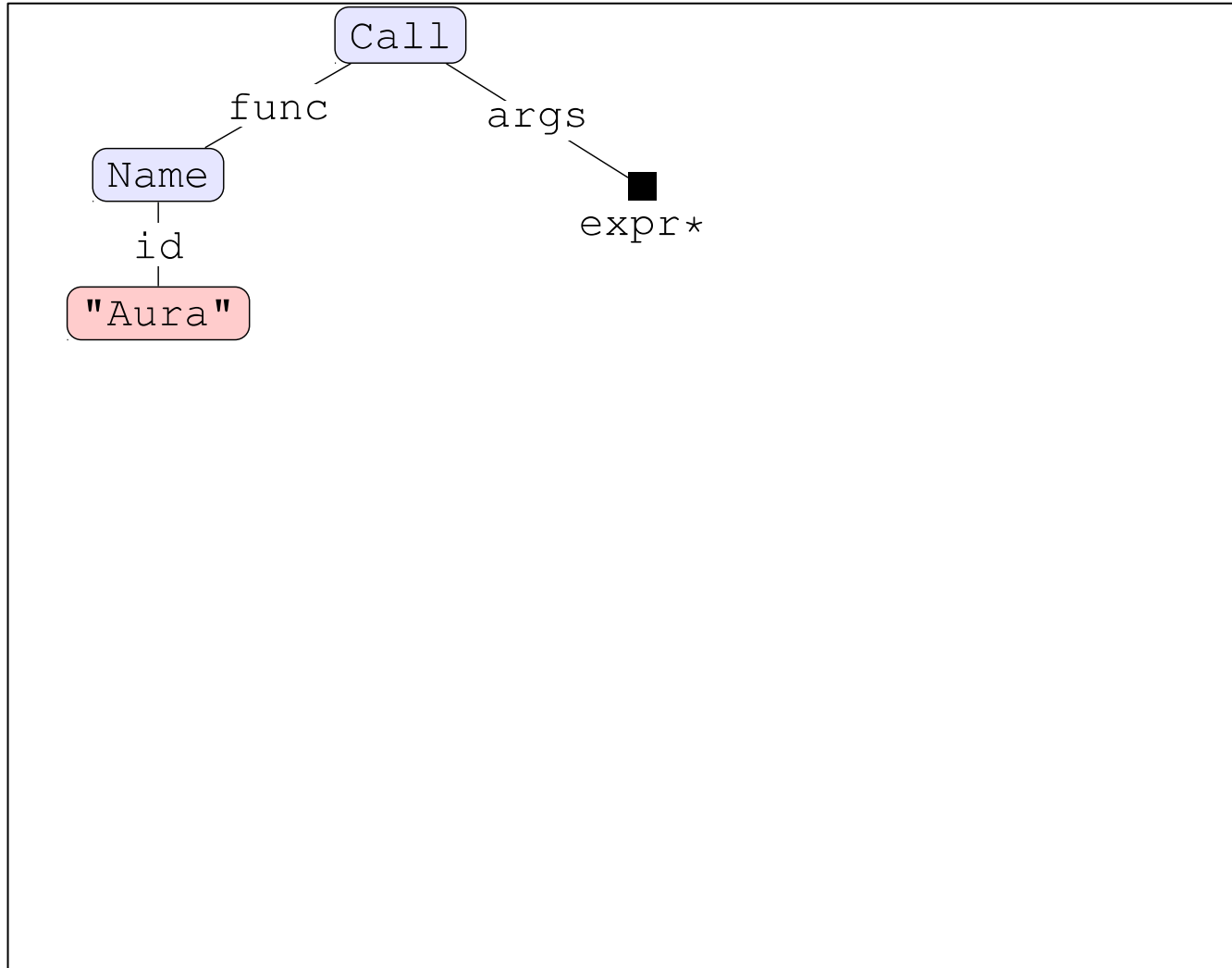


Decoding Process



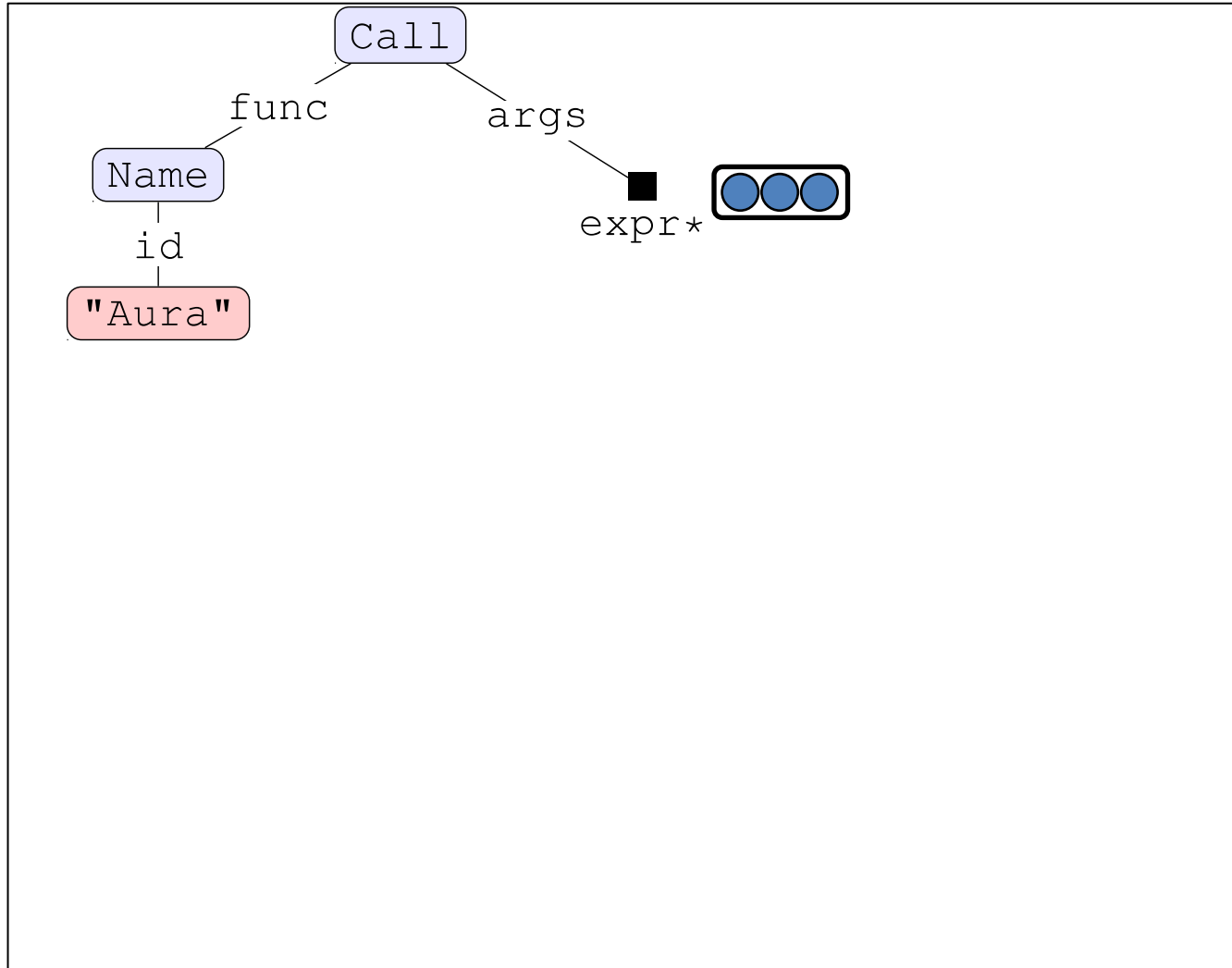


Decoding Process



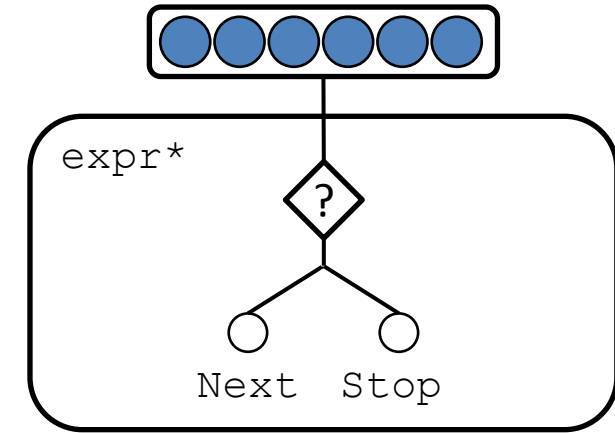
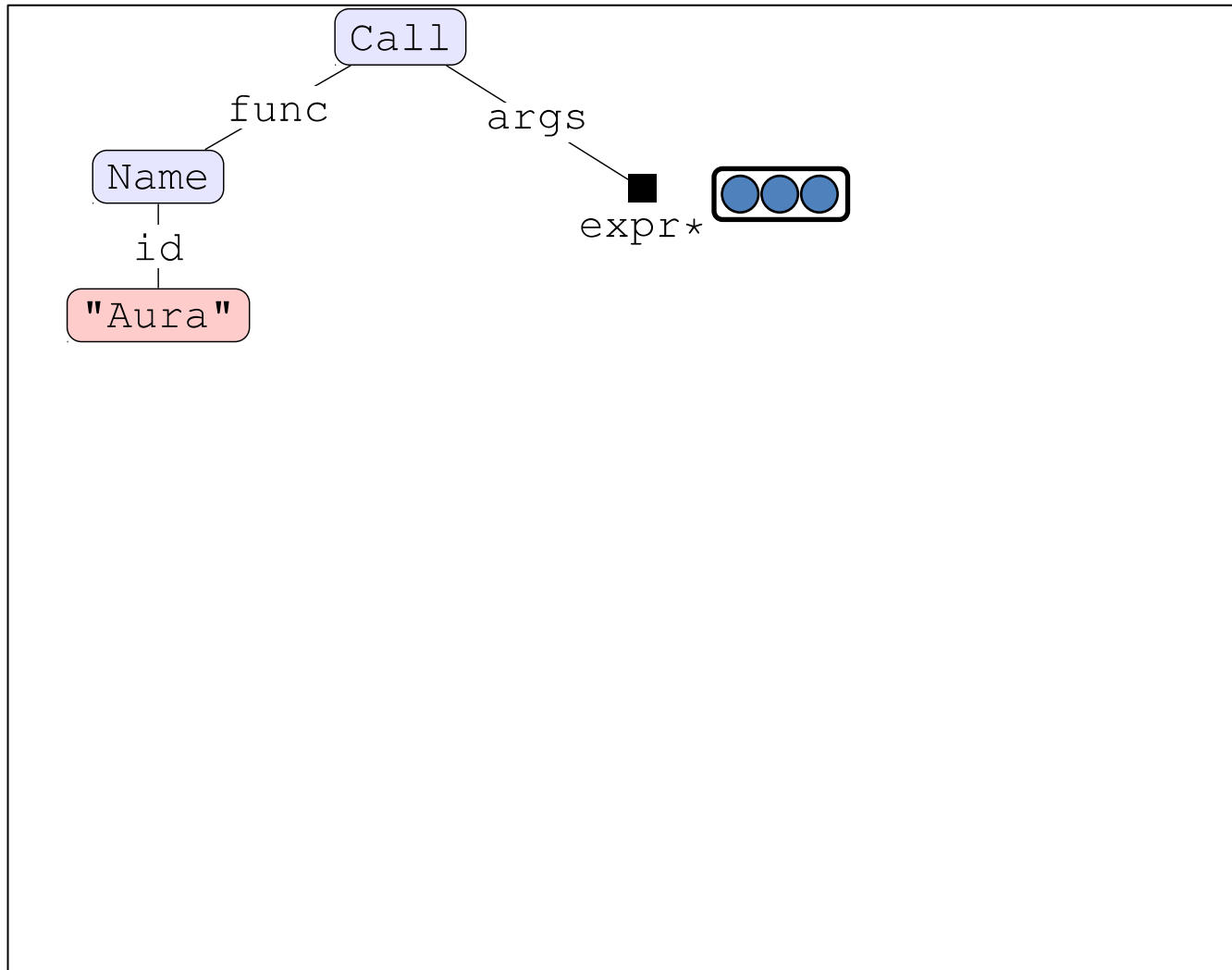


Decoding Process



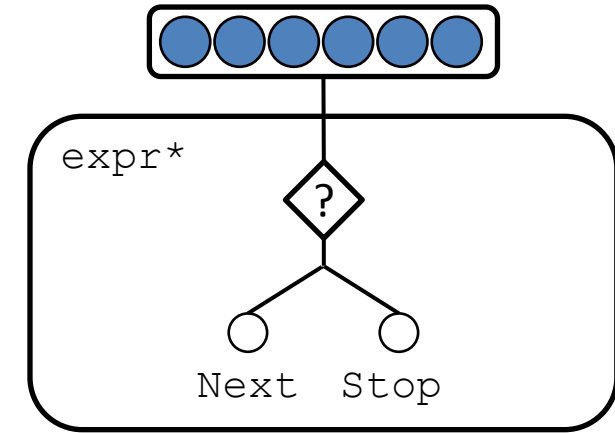
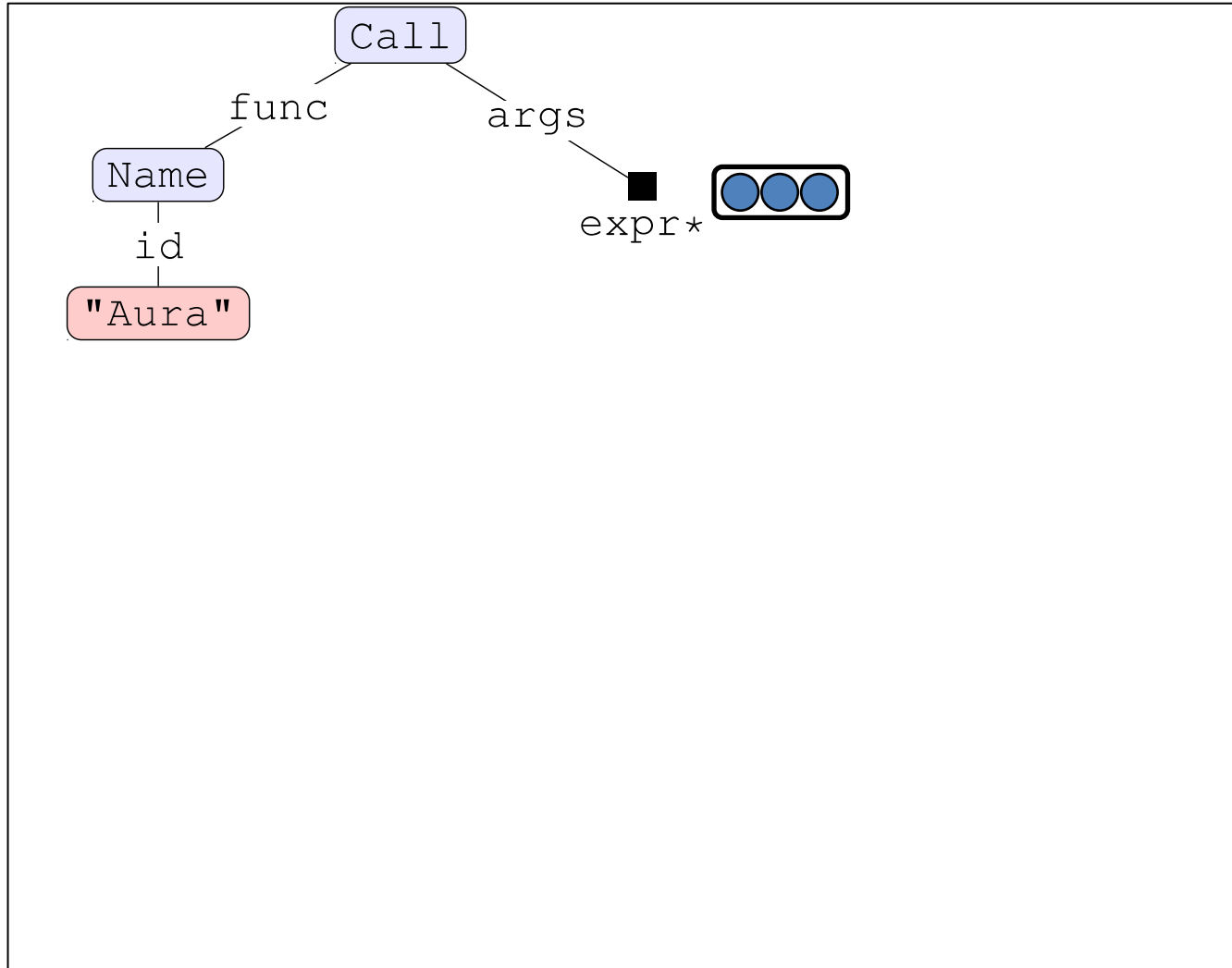


Decoding Process



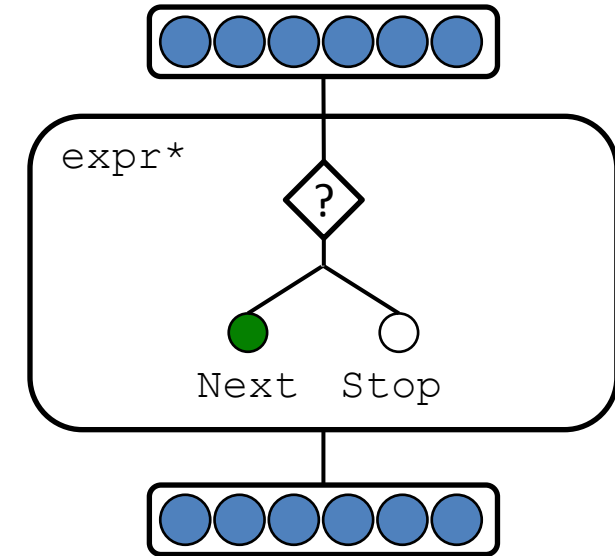
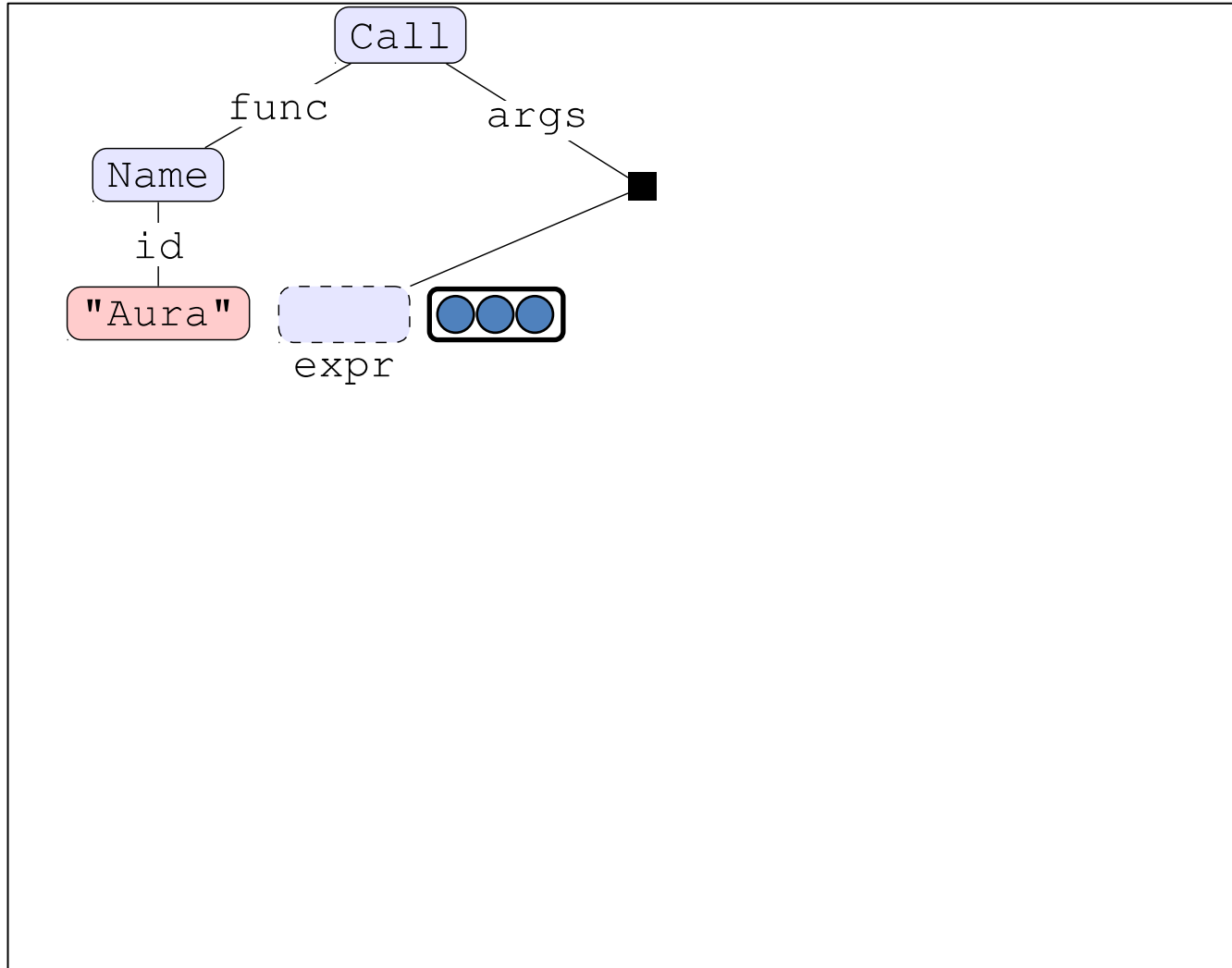


Decoding Process



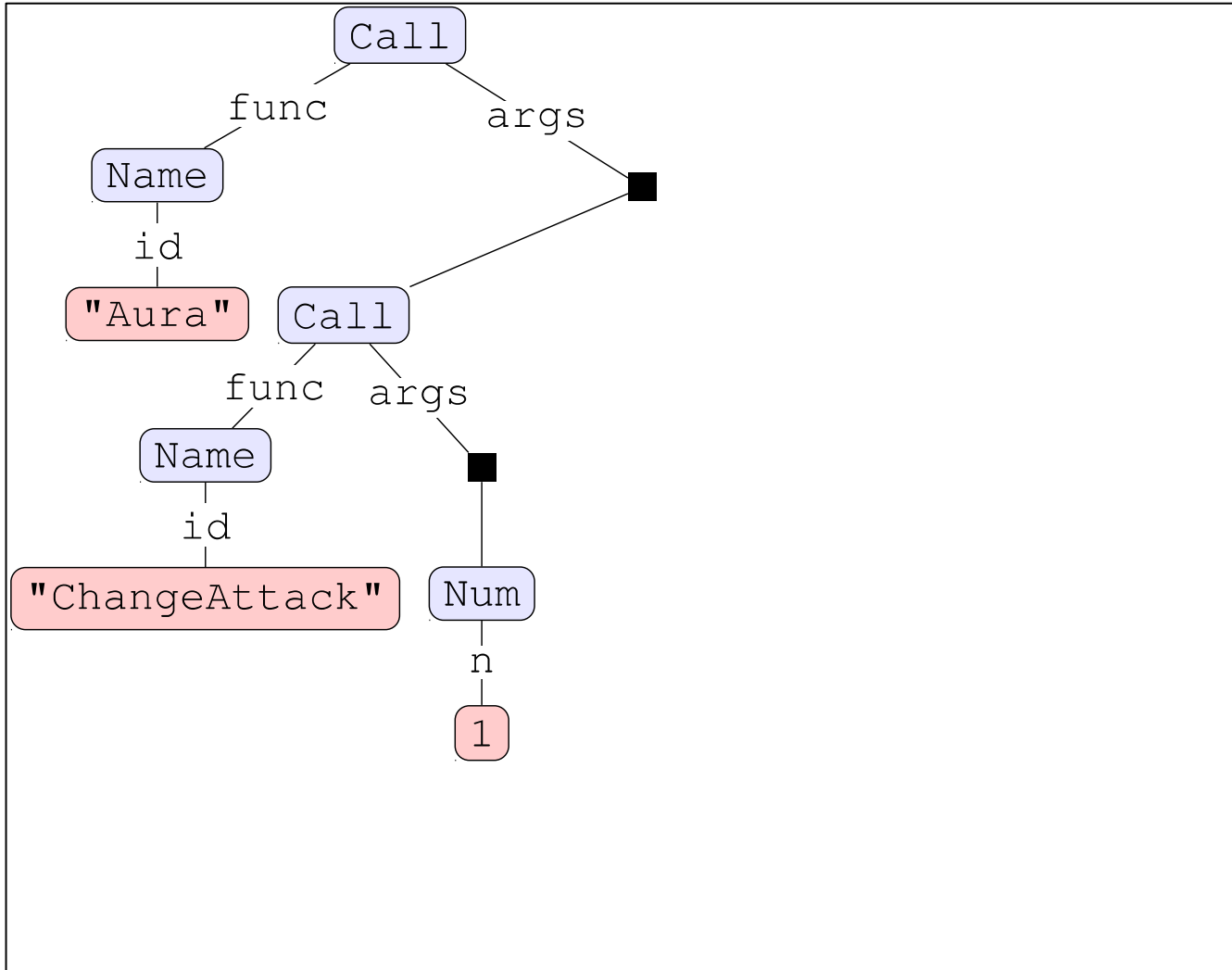


Decoding Process



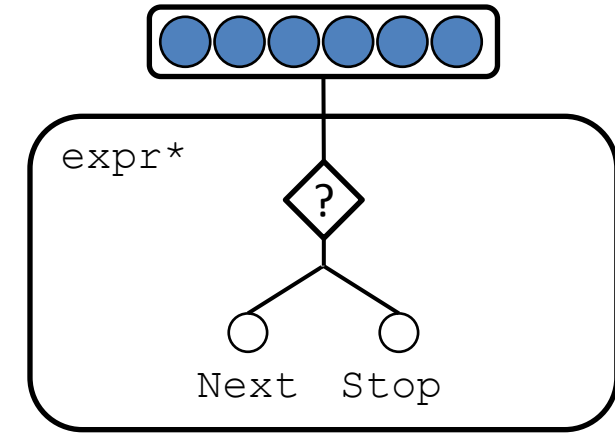
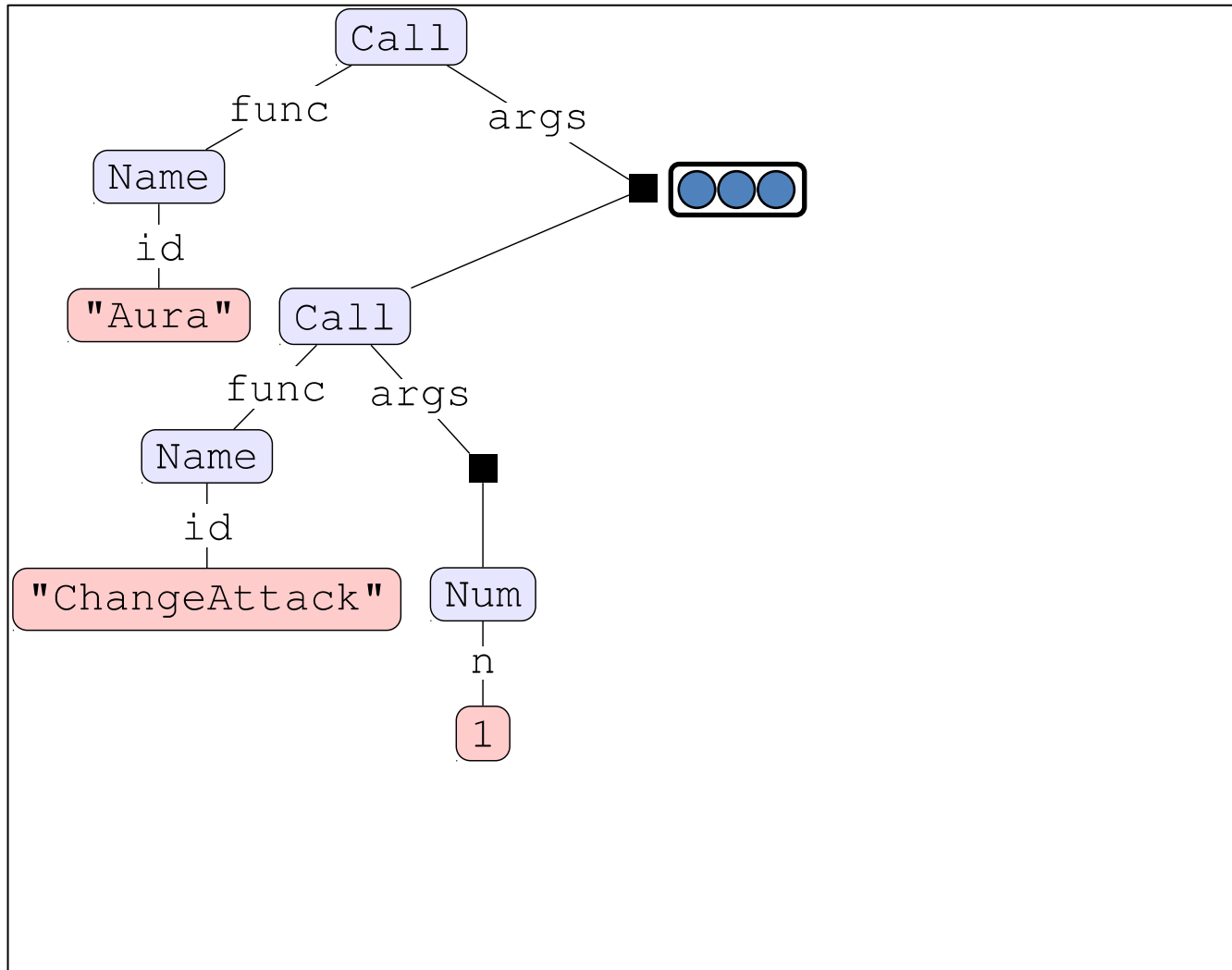


Decoding Process



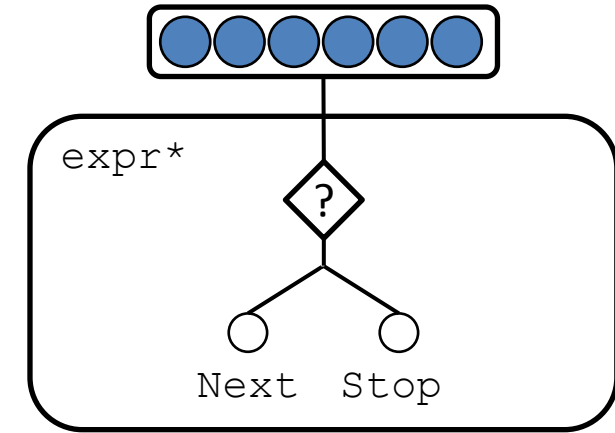
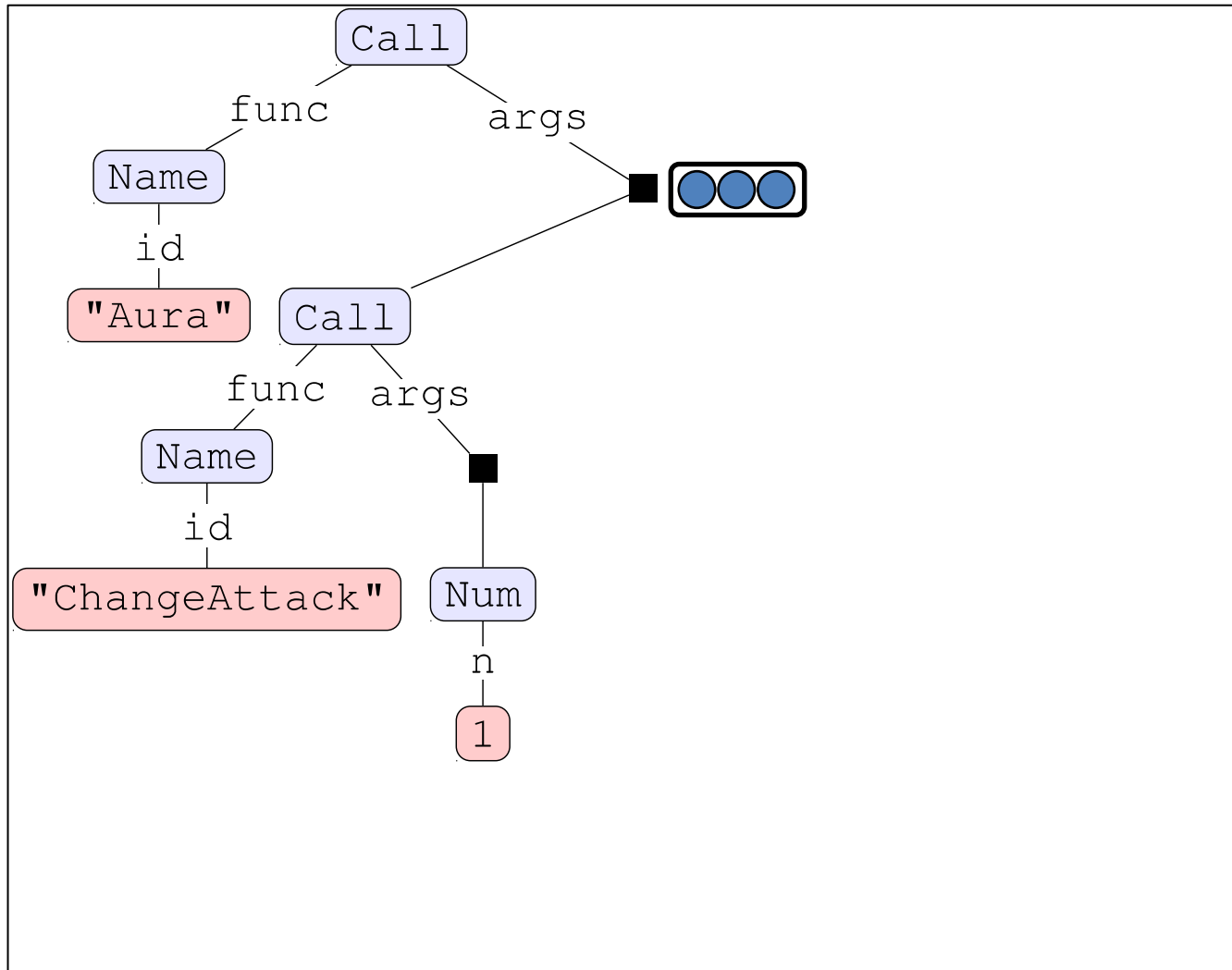


Decoding Process



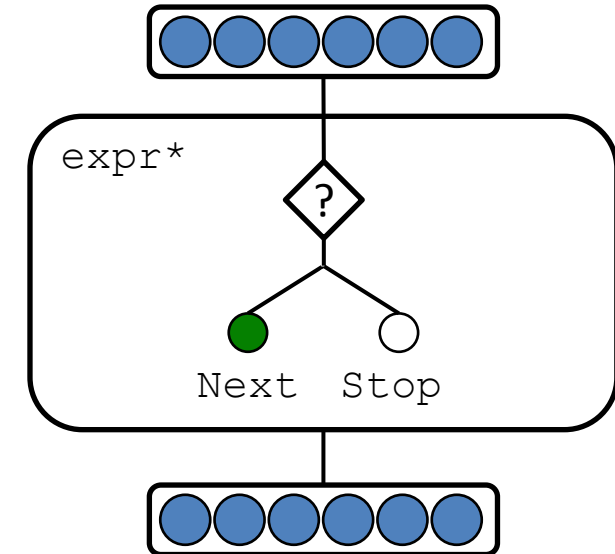
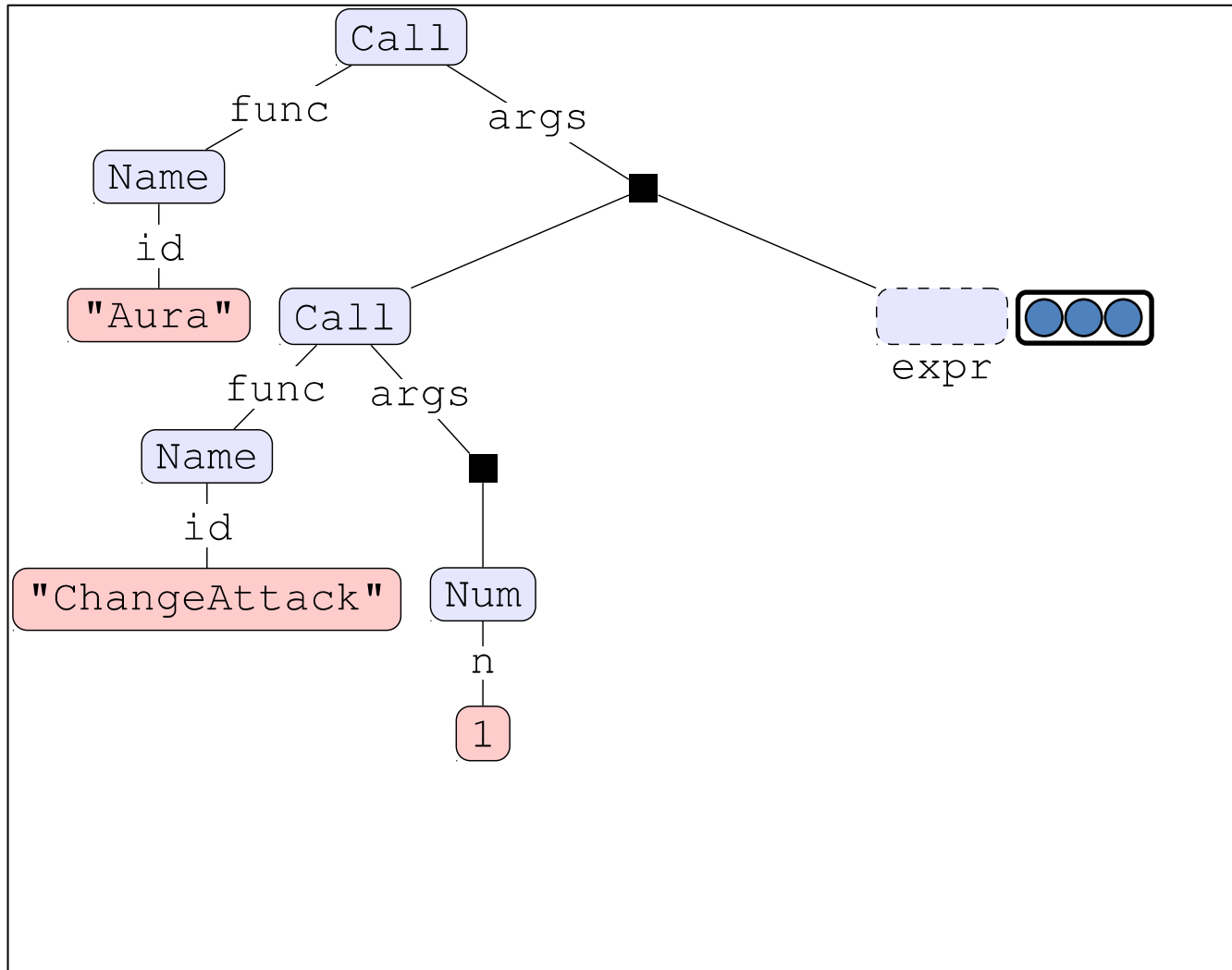


Decoding Process



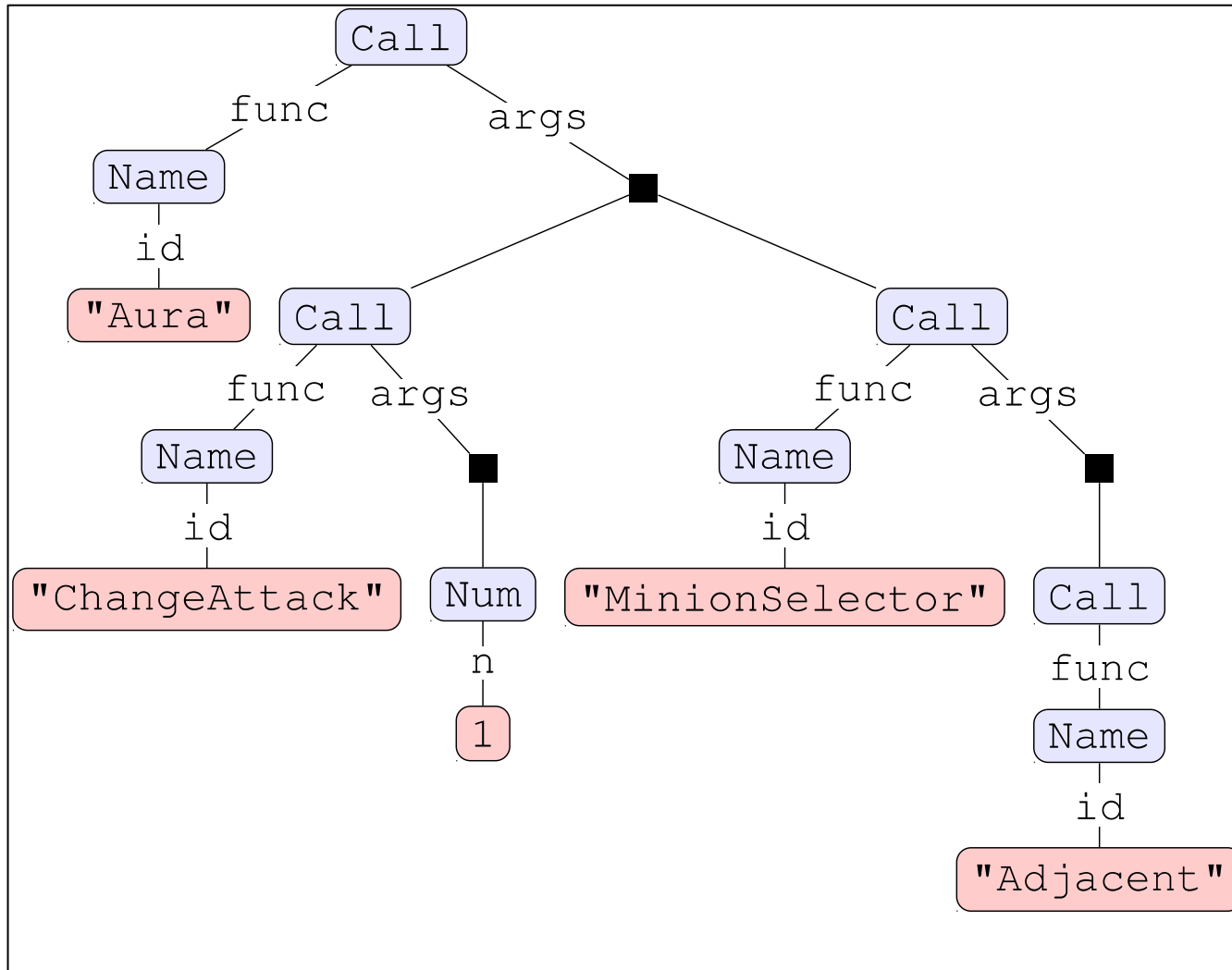


Decoding Process



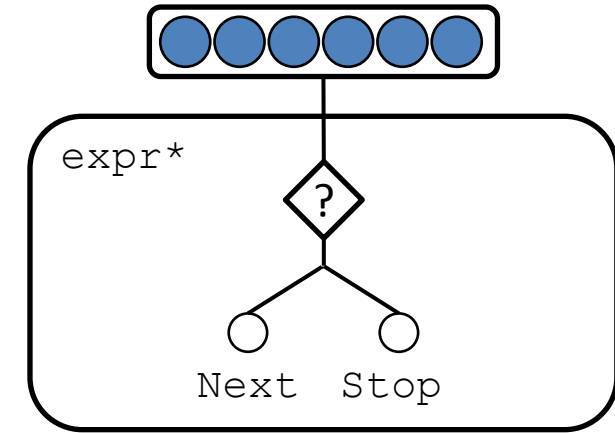
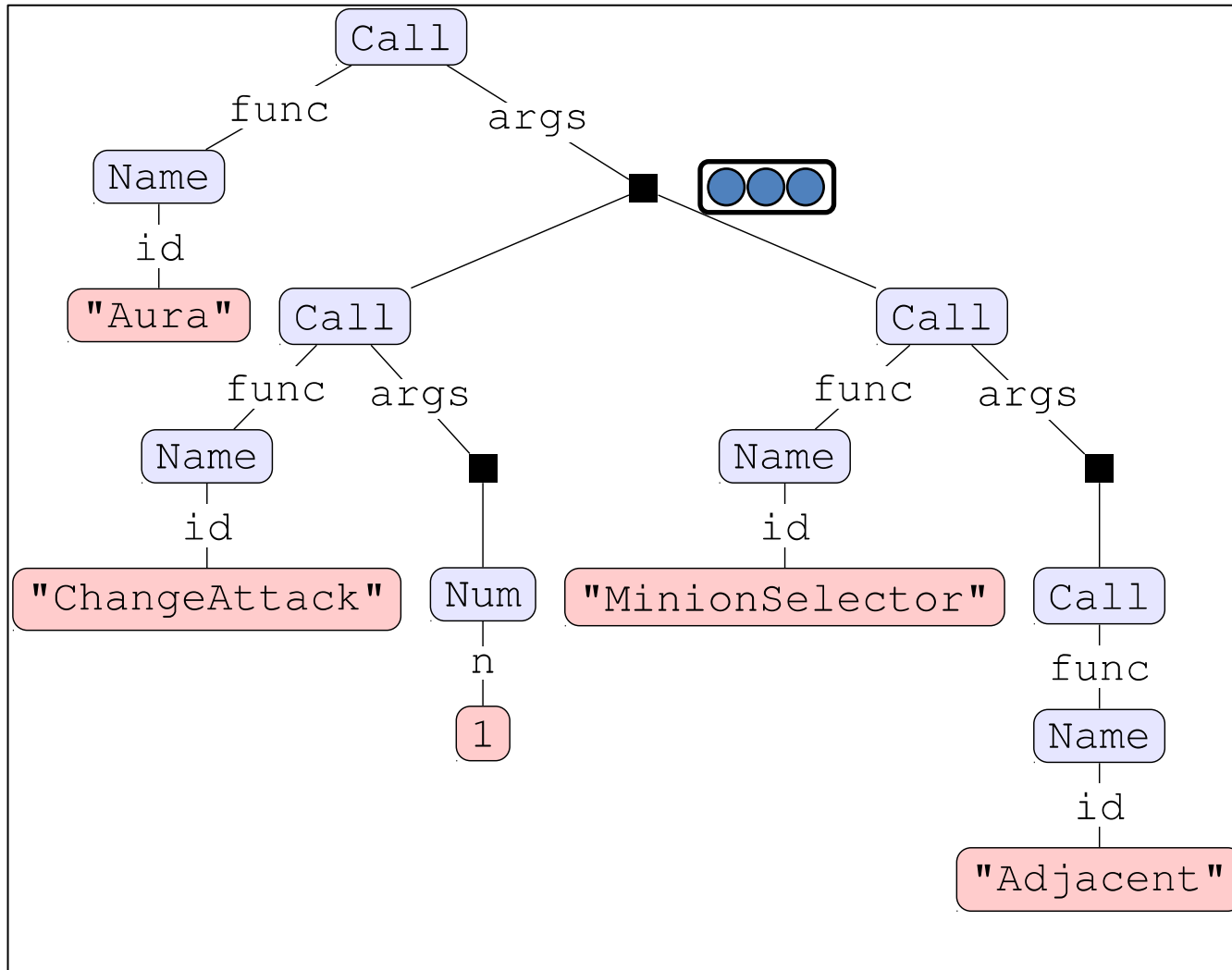


Decoding Process



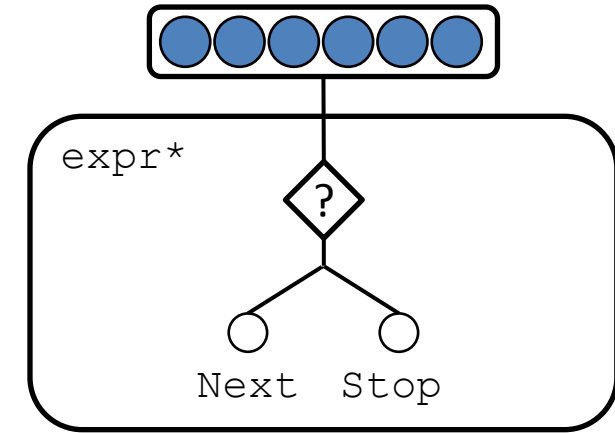
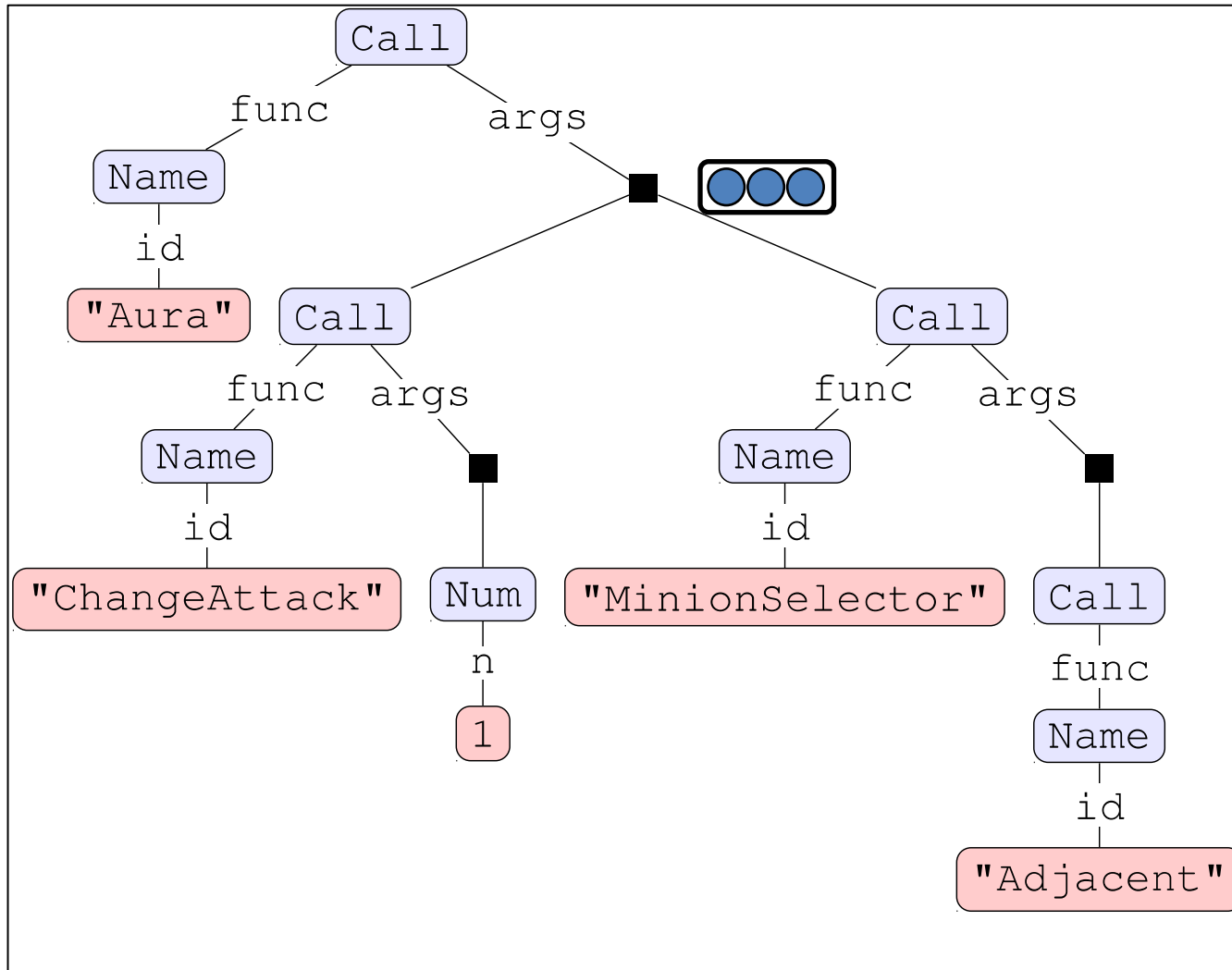


Decoding Process



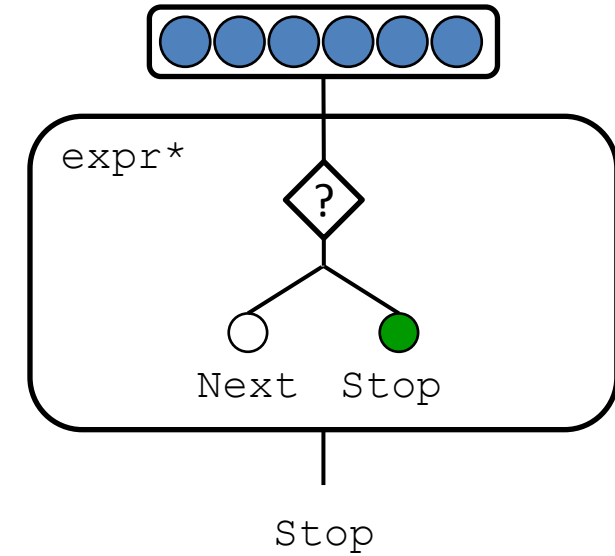
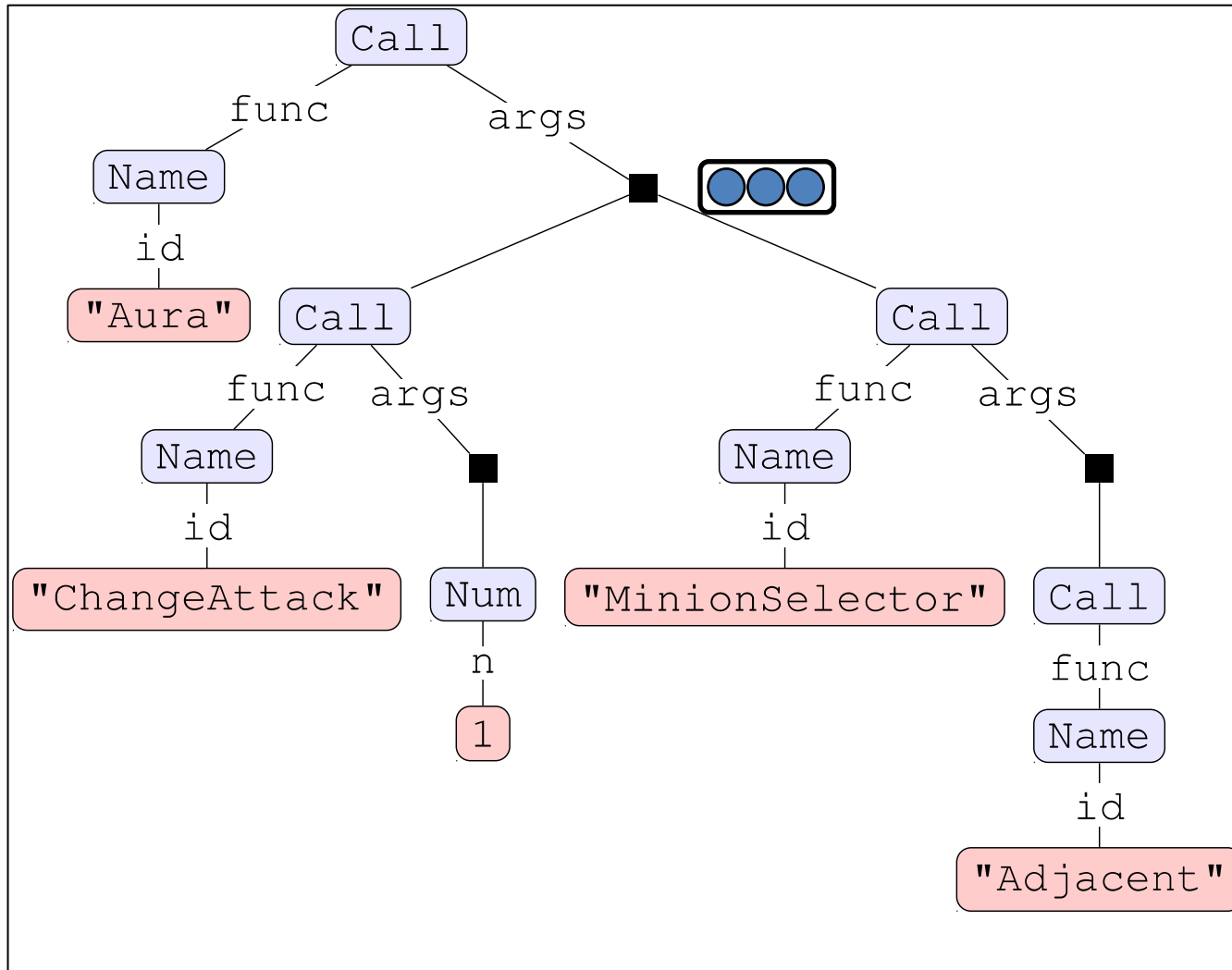


Decoding Process



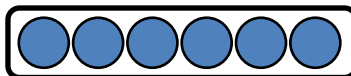


Decoding Process



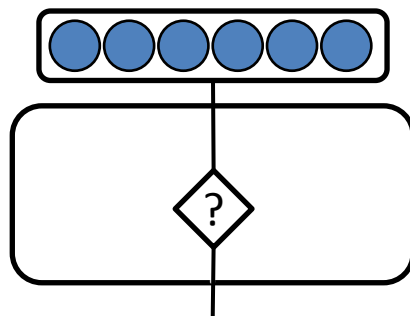


Training



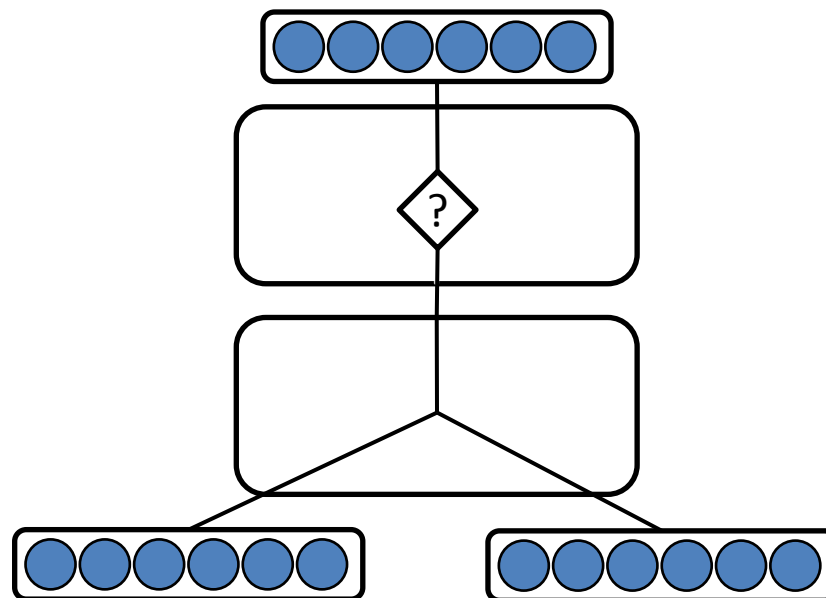


Training



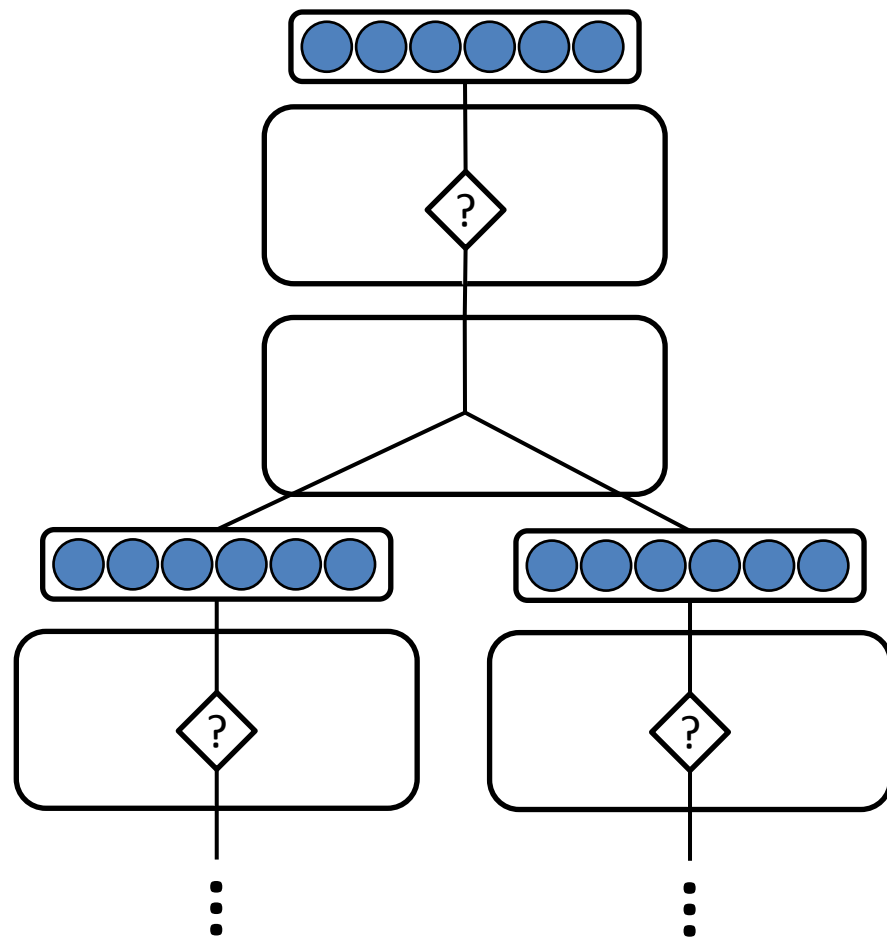


Training



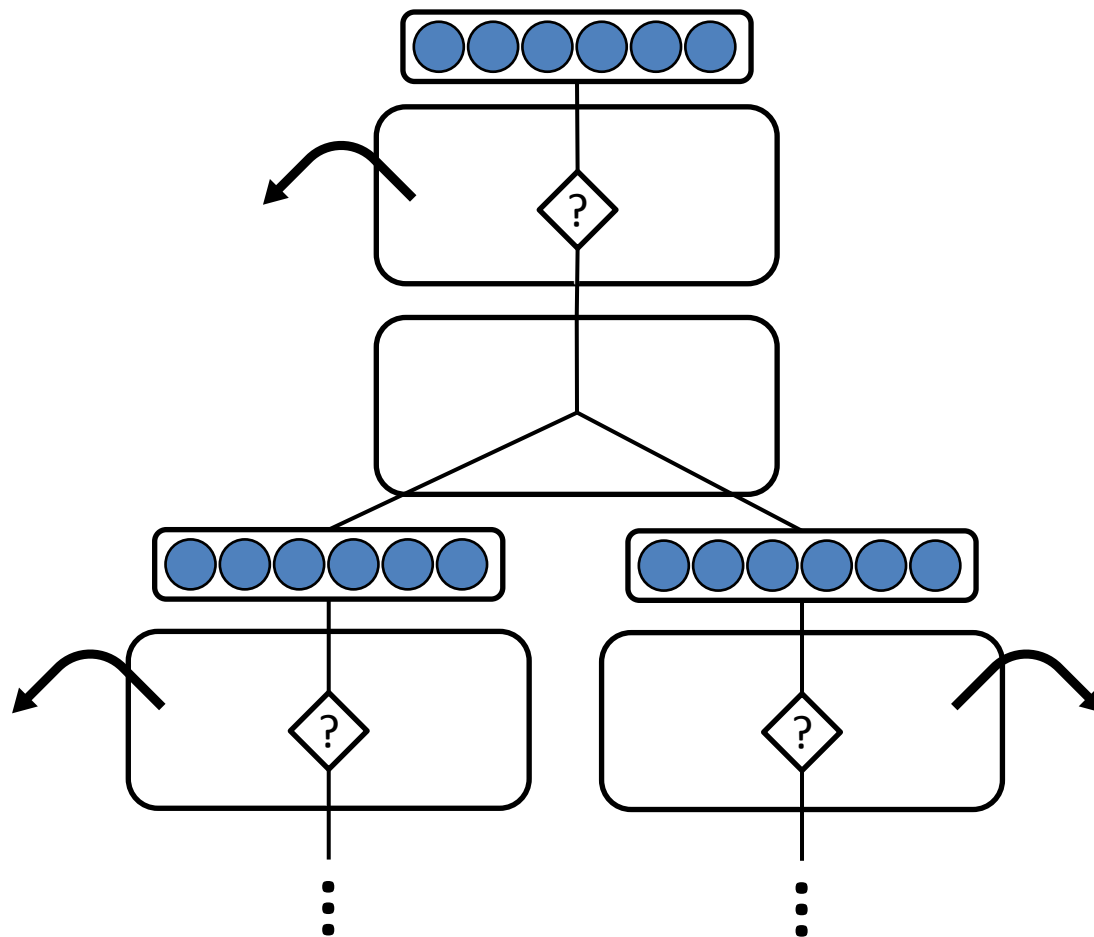


Training





Training

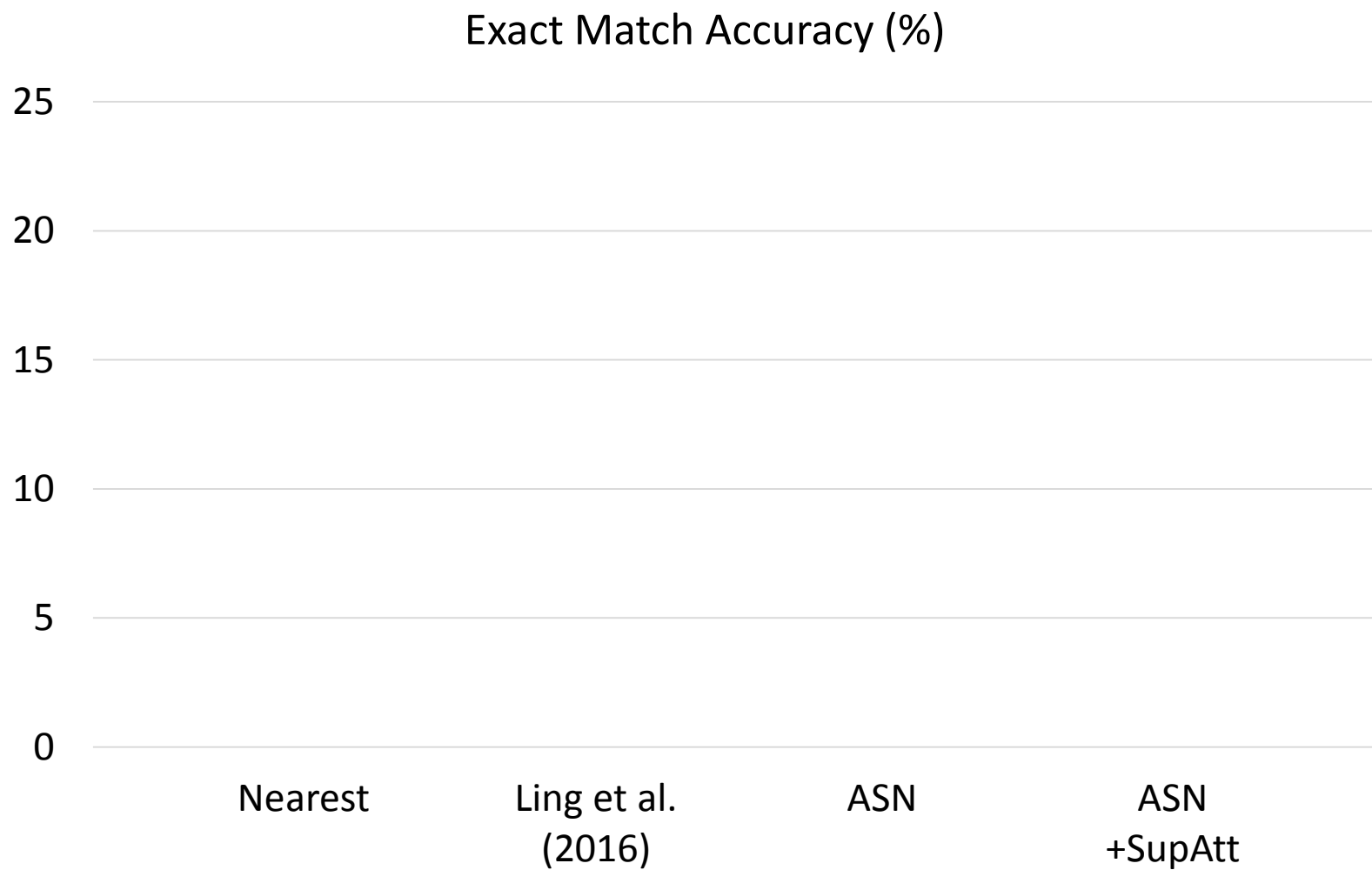




Hearthstone Results

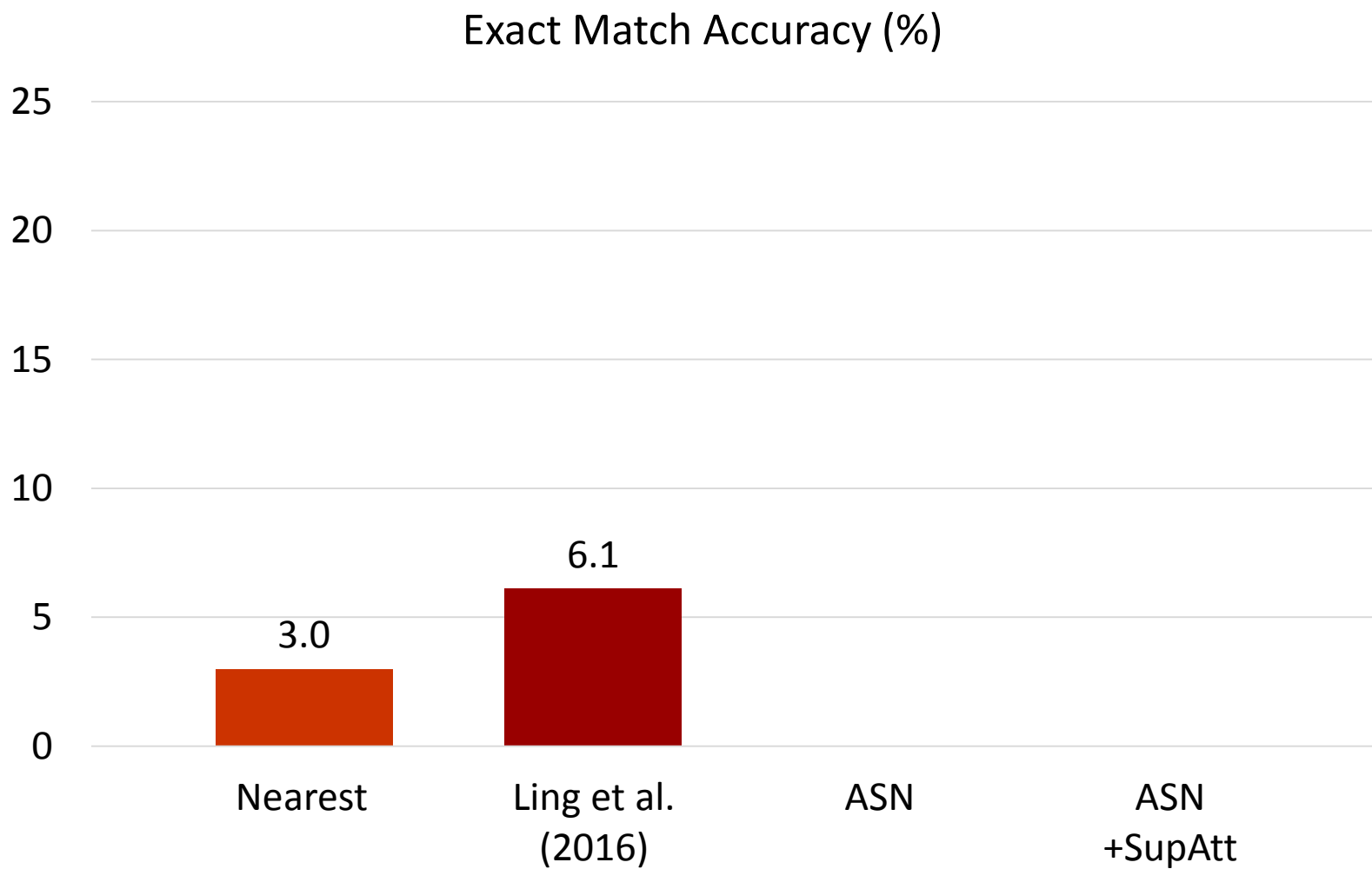


Hearthstone Results



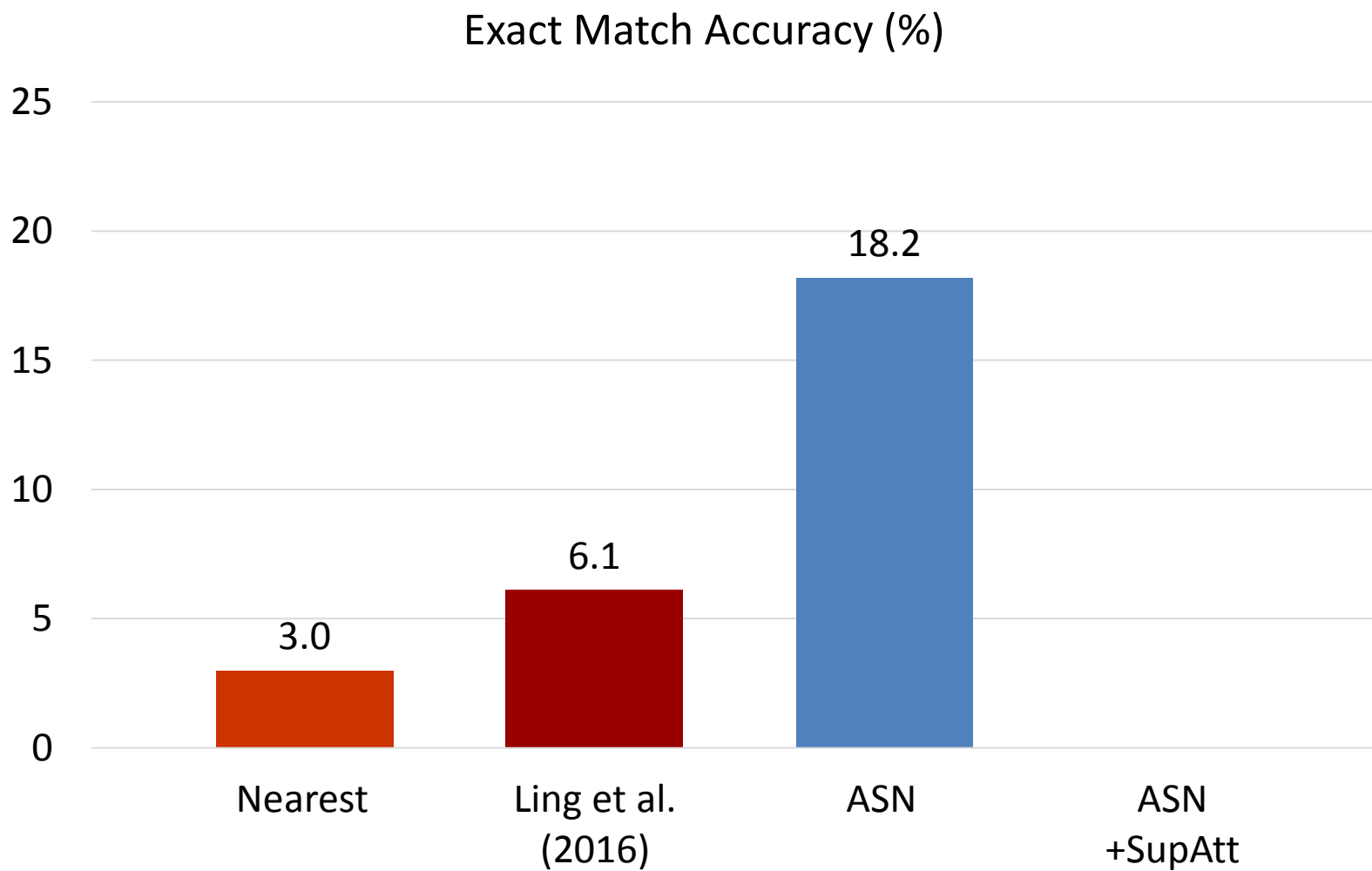


Hearthstone Results



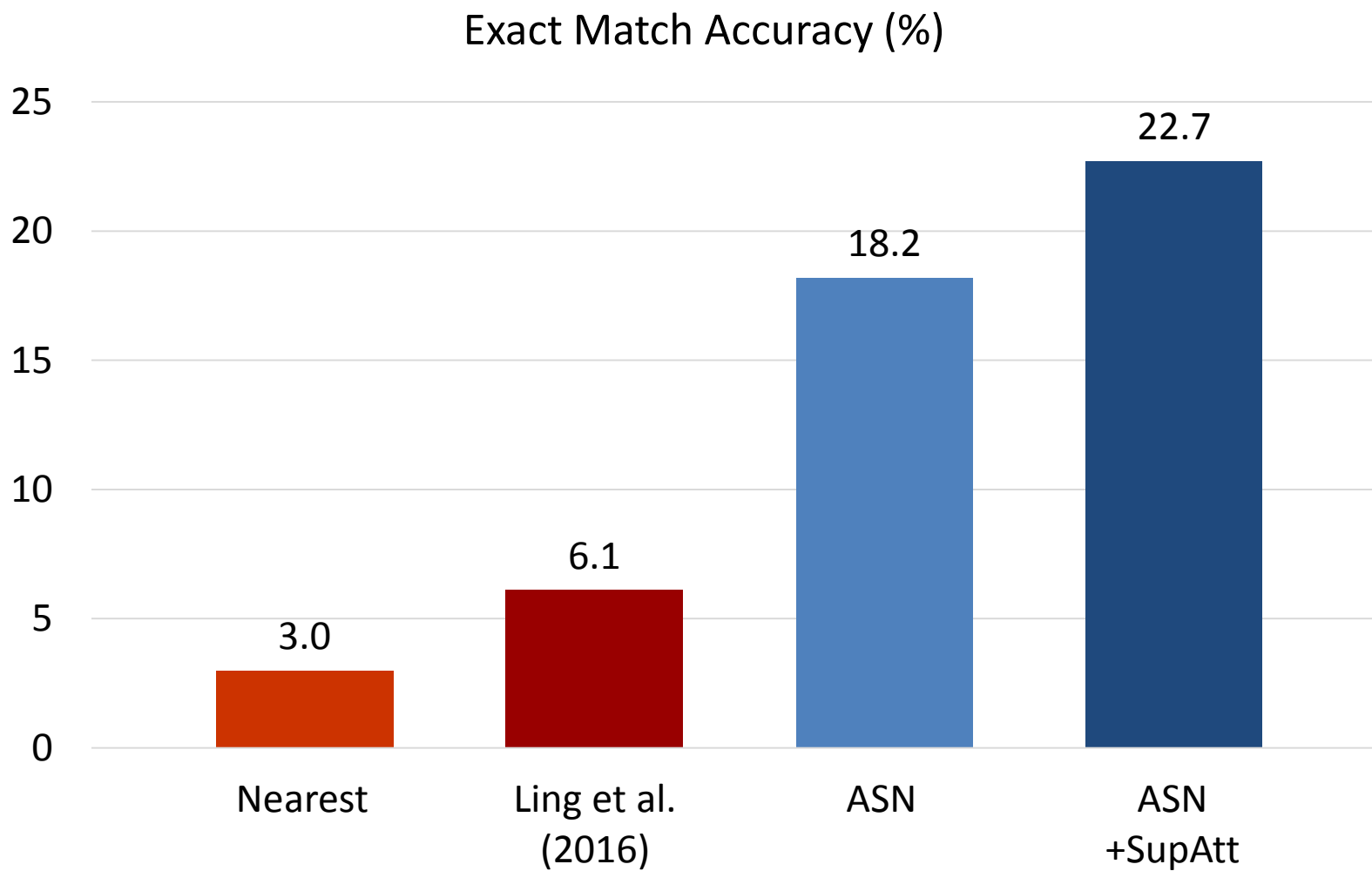


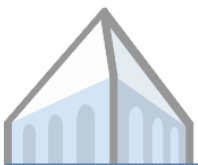
Hearthstone Results



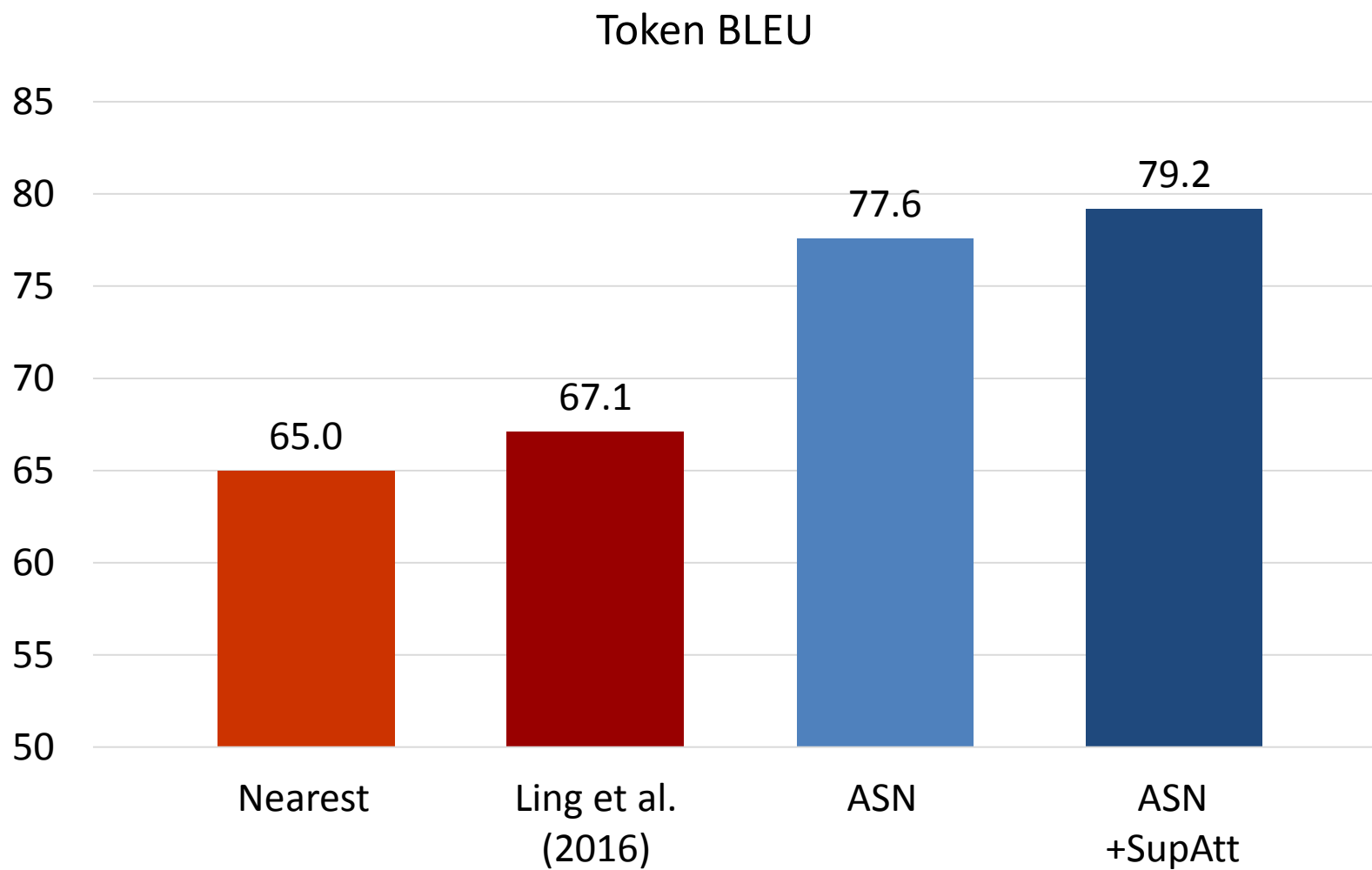


Hearthstone Results



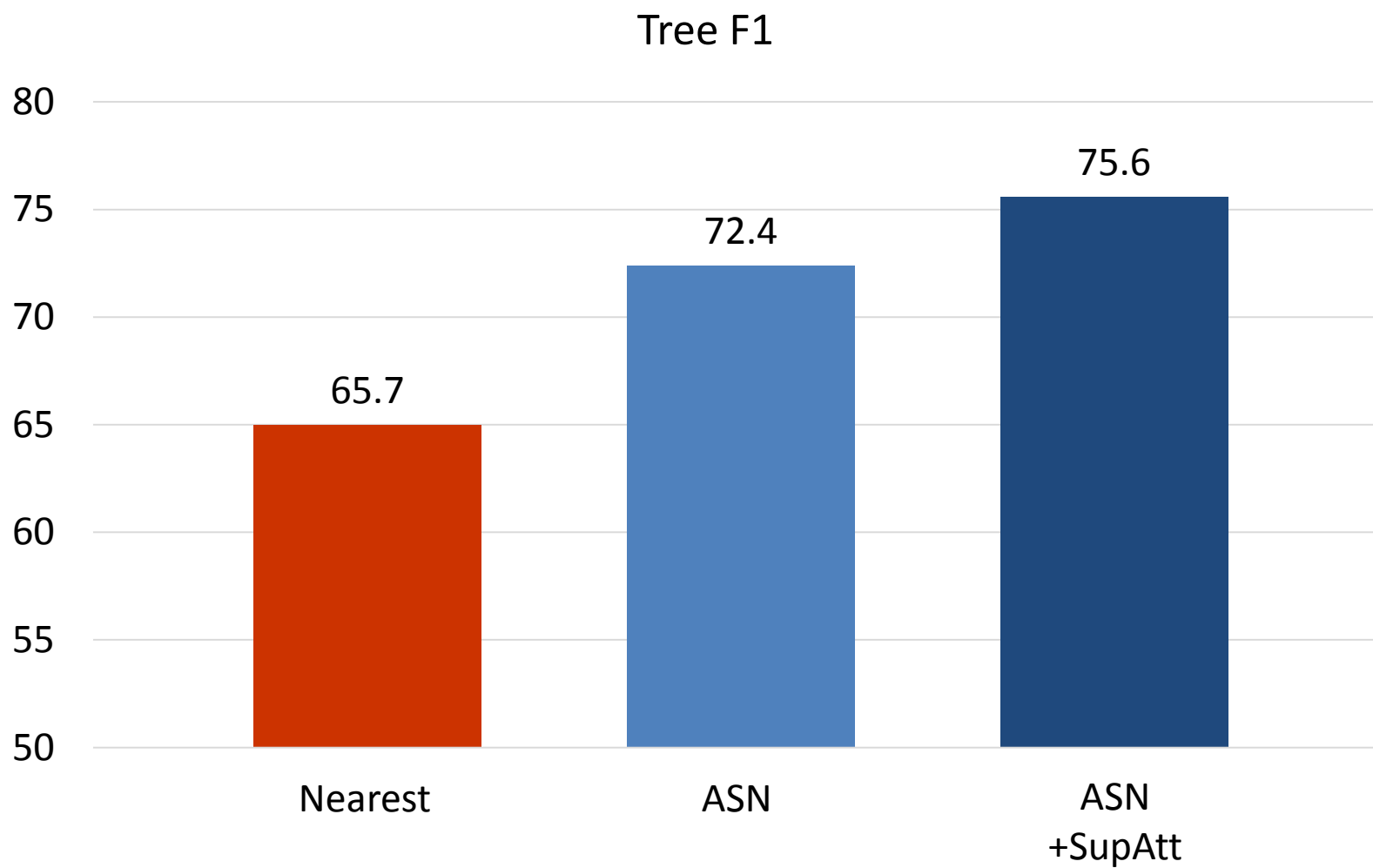


Hearthstone Results





Hearthstone Results





Semantic Parsing

show me the fare from ci0 to ci1

$\lambda x. \exists y. \text{from}(y, \text{ci0}) \wedge \text{to}(y, \text{ci1}) \wedge \text{equals}(\text{fare}(y), x)$



Semantic Parsing Results

Geo



ATIS



Jobs





Conclusion

Code generation and similar systems benefit from:

- Representing structure in output space
- Modular network architectures



Thanks!