What is a Norm?

A characterization of the normal

- Something accepted by many
- A standard
  - Formal or *de jure*
  - Informal or *de facto*

Sociocognitive Basis of Norms

Muzafer Sherif, 1933

- Autokineisis
- Confederate study
- Willingness to shift perception

What is a Norm?

Emphasis in this course

- A *rule of encounter* agreed to by the participants
- An elementary directed relationship between two parties
  - An expectation
  - Moreover, a legitimate expectation
- Stronger than a mere convention

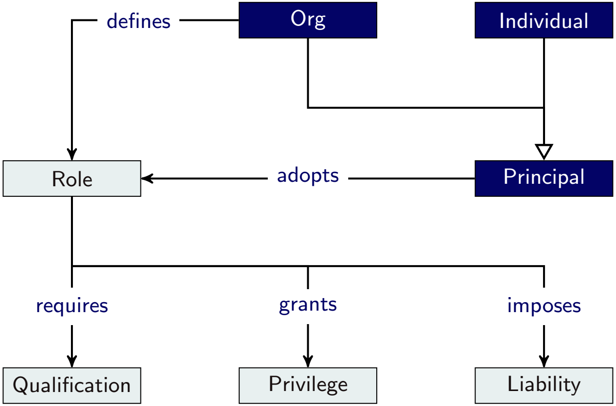
Must be violable!

Achieving Governance: Agents and Orgs

Put collaboration center stage

- Agents (including Orgs) are active computational entities, and represent principals
- Agents represent the stakeholders: people and organizations
  - Provide a locus for interaction
- Orgs are like *institutions:* have an identity and life time distinct from their members; also modeled as agents
  - Examples: NCSU, UNC System, . . .
  - Provide a locus for roles and authorizations
  - Enforce behavioral constraints on members
    - Their main hold over their members is the threat of expulsion

Governance Conceptually



Norms Capture the Social State of an STS

Norms capture an STS in an implementation-independent way by specifying its roles'

- Qualifications
- Privileges
- Liabilities

Norms inform decision-making

- In making accountability judgments, e.g., for determining sanctions
- For maximizing private gain, e.g., by appropriately scheduling actions
- For determining others' trustworthiness

Legal Concepts: Oldest Model of Norms

Inherently multiagent: about interactions among autonomous parties

- Directed obligations
- One party being obliged to another party
- Multiagent flavor
- Contrast with traditional deontic logic
  - Zero-agent: it is obligatory that ...
  - One-agent: you are obliged to do ...

Rights

- The rights or claims a party has on another party
  - Not the right (ethical) thing to do
- The claims of one party are the duties of another: claim is a correlate of duty

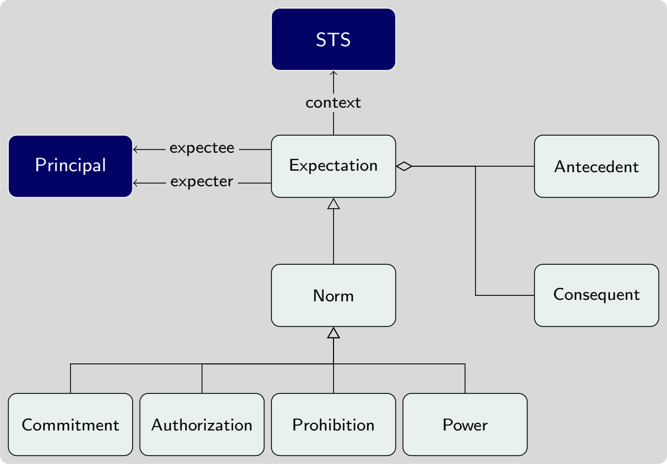
Hohfeldian Concepts: 1

- The term right is used ambiguously
- Sixteen concepts distinguish the main situations:
  - Four main concepts
  - Their correlates
  - Their negations
  - Their negations' correlates

Hohfeldian Concepts: 2

- Claim-duty: as above
- Privilege-exposure: freedom from the claims of another agent
- Power-liability: when an agent can change the claim-duty relationship of another agent
- Immunity-disability: freedom from the power of another agent

Express Legitimate Expectations as Norms



Norms are Directed and Characterize Accountability

Norm Type	Accountable (Expectee)	Privileged (Expecter)
Commitment	Debtor	Creditor
Prohibition	Authorizer	Authorizee
Authorization	Authorizer	Authorizee
Power	Empowering	Empowered

Juggling Contract: 1

Contract For: Canterbury Centre Dinner 2003 ("CCD"), Friday 6 June 2003, 24 High Street, Canterbury

This agreement is entered into between the University Juggling Club ("UJC") and the Canterbury Centre Dinner 2003 on the following terms:

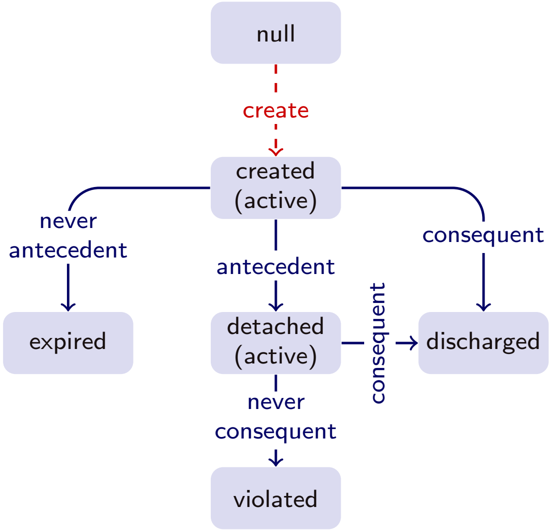
1. Service Provider: University Juggling Club.
2. Employer: Canterbury Centre Dinner.
3. To be provided by UJC: Performers: J Woods (juggler); one other juggler; all equipment necessary for performance.
4. To be provided by CCD: Cloakroom.
5. Venue address: 24 High Street, Canterbury.
6. CCD understands that performances are restricted in venues with ceilings of insufficient height. The ideal height is 5 meters. Outside performances are restricted in rain or strong winds.
7. Date of Performance: Friday 6 June 2003, starting at 6:30PM.
8. Duration of Performance: 1.5 hours. Short (less than one minute) breaks are part of the performance.

Juggling Contract: 2

9. Fee: £30 per juggler + £10 expenses + £90 insurance (total £160).
10. If UJC is forced to cancel, all monies (including £90 deposit) will be refunded in full. If the Employer cancels with at least 24 hours notice, UJC will retain £90 and return any other monies.
11. Should poor weather mean that the Event takes place indoors, UJC will refund £10 expenses.
12. Performers will not consume any alcohol until after completion of services as agreed.
13. CCD will be responsible for compensation to UJC for damage to equipment caused by those attending the Event unless damage is caused when (if) Performers have left equipment unguarded.
14. UJC will be liable for any injury sustained by a guest at the Event if such injury results from provision of services as agreed upon in this contract unless the Event fails to provide a suitable area for performance.

Lifecycle of a Commitment Instance

The transitions are logic-based, except create, which is social (see later)



Example: Commitment Progression

C(Buyer, Seller, goods, pay): created

- If *goods* $\wedge$ C(Buyer, Seller, goods, pay) Then
  - Active and detached (aka unconditional or base)
  - C(Buyer, Seller, T, pay)
- If C(Buyer, Seller, T, pay) Then
  - If *pay* Then Satisfied
  - If never *pay* Then Violated
- If C(Buyer, Seller, goods, pay) Then
  - If *pay* Then Satisfied
  - If never *pay* and never *goods* Then Expired

Can be nested:
C(Seller, Buyer, pay, C(Shipper, Buyer, T, deliverGoods))

Example: Commitment Progression is Piecemeal

Showing a few paths

C(Seller, Buyer, pay $\vee$ coupon, tea $\wedge$ scone): created

- If *tea* $\wedge$ C(Buyer, Seller, goods, pay) Then
  - C(Buyer, Seller, pay $\vee$ coupon, scone)
- If C(Buyer, Seller, pay $\vee$ coupon, scone) Then
  - If *scone* Then Satisfied
  - If never *scone* Then $\langle$no change$\rangle$
- If C(Buyer, Seller, pay $\vee$ coupon, scone) Then
  - If *pay* Then C(Buyer, Seller, T, scone)
- If C(Buyer, Seller, T, scone) Then
  - If *scone* Then Satisfied
  - If never *scone* Then Violated

Operations on Norms

Via explicit operations or because of logical properties

Logic-based: As above

- *detach* brings it into force
- *discharge* ends it in a "good" state
- *violate* ends it in a "bad" state

Social: To be revisited under communication

- *create* establishes the commitment
- *cancel*: the accountable party backs out
  - May lead to sanctions
- *release*: the party with standing lets go
  - Canonical example: declining an offer
- *delegate*: the accountable party makes another party accountable
  - The original party may remain on the hook too
- *assign*: the party with standing passes on the standing to another party

Cupid: Event-Based Norms Language

Commitments, Authorization, Prohibition, and Power

- Systematic treatment of norm lifecycles
- Features needed for real-world scenarios
  - Relative deadlines
  - Each event has a time of occurrence
  - Nested norms
  - Complex events
    - Expressed via Boolean operators and nesting
    - Immutable
    - Computing the time when a complex event occurs is nontrivial

An Information Model and Commitment Specification

E-commerce setting

```
schema
Quote(S, B, ID, item, uPrice) with key ID
Accept(B, S, ID, qty, addr) with  key ID
Payment(B, S, ID, pPrice) with  key ID
Shipment(S, B, ID, addr) with key ID
Refund(S, B, ID, rAmount) with key ID
```

```
commitment DiscountQuote S to B
create Quote
detach Accept[, Quote + 4] and Payment[, Quote + 4]
where pPrice >= 0.9 * uPrice * qty
discharge Shipment[, detached DiscountQuote + 4]
```

Query Results

For times up to 16 June 2025

Quote

ID	item	uPrice	t
T1	fig	1	1 June 2025
T2	pear	1	1 June 2025

Accept

ID	qty	addr	t
T1	1	Lancaster	2 June 2025
T2	1	Raleigh	2 June 2025

Payment

ID	pPrice	t
T1	1	2 June 2025
T2	1	2 June 2025

Shipment

ID	addr	t
T1	Lancaster	3 June 2025

discharged

ID	t
T1	3 June 2025

violated

ID	t
T2	7 June 2025

Example: Compensation

A commitment that refers to the state of another commitment

```
commitment Compensation S to B
create Quote
detach violated(DiscountQuote)
discharge Refund[,violated(DiscountQuote) + 9] where
  rAmount = pPrice
```

Canonical Queries for DiscountQuote

For created, detached, expired, discharged, and violated instances

- Generated SQL is long and complicated; near impossible to write manually
  - The query for *violated DiscountQuote* is 413 lines long
  - The five queries amount to 1060 lines

Cupid Specifications

Scenario: Healthcare wearables and privacy

```
schema
Registered(pID, jID, resID, council) key resID
Signedup(pID, hID, accID) key accID
Allowed(pID, hID, discID, accID, tpID, info) ket discID
Revoked(pID, hID, discID) key discID
...
```

```
commitment DisclosureCom hID to pID
create Signedup
detach Allowed except Revoked
discharge created DisclosureAuth[0, Allowed+1]
```

Cupid: Authorization and Prohibition

```
authorization DisclosureAuth tpID by hID
create SentCred
detach ReqData
discharge Accessed[ReqData+1, ReqData+10]
```

```
prohibition DisclosureProh tpID by hID
create discharged DisclosureAuth
violate Forwarded
```

Cupid: Power and Commitment Involving Aggregation

```
power ConsentPower pID by jID
create Registered
detach Signedup[Registered ,]
discharge Allowed[Signedup ,] and Revoke[Signedup ,]
```

```
commitment SanctionCom hID to pID
create Signedup
detach count forID of violated DisclosureProh as numViol
  group by discID > 2 [Signedup, Signedup+365]
discharge Sanction[, detached SanctionCom+10] where
  details="Out of compliance"
```

Norm Query: *discharged DisclosureAuth*

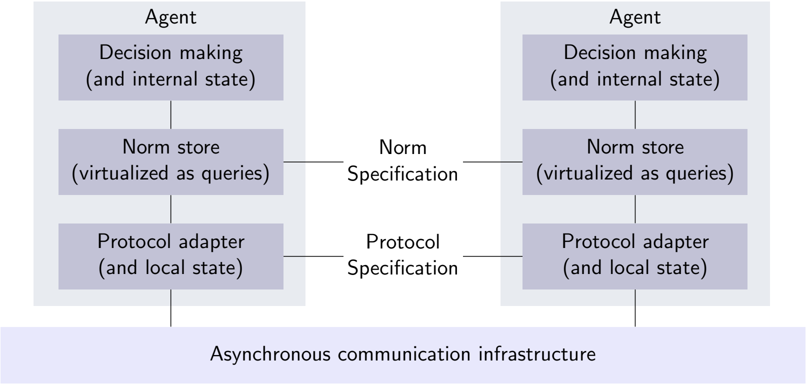
Generated SQL Query (65 lines) returns all discharged instances of DischargeAuth

```
SELECT
hID, tpID, discID, credentials, reqID, response, t
FROM
(SELECT
hID, tpID, discID, credentials, reqID, response,
GREATEST(Query21.t, Query28.t3) AS t
FROM
(SELECT
hID, tpID, discID, credentials, t
FROM
SentCred) AS Query21

...


NATURAL JOIN (SELECT
hID, tpID, discID, reqID, request, t AS t6
FROM
(SELECT
hID, tpID, discID, reqID, request, t
FROM
ReqData) AS Query24) AS Query36
WHERE
'1000-01-01 00:00:00' <= Query22.t
AND Query22.t < Query36.t6 + INTERVAL 10 DAY) AS Query31) AS Query32) AS Query25)
          AS Query28) AS Query26
WHERE
t < NOW();
```

Decentralized Architecture

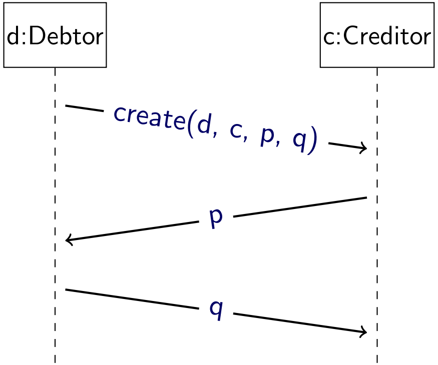
Norms and protocols model the interactions

- Causing an event: Upon sending a message
- Observing an event: Upon sending or receiving a message
- Norm lifecycle: computed from the information



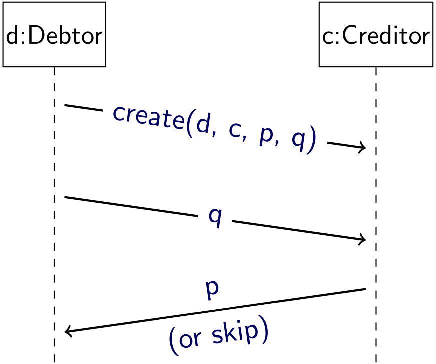
Operationalizing Commitments: Detach then Discharge

C(debtor, creditor, antecedent, consequent)



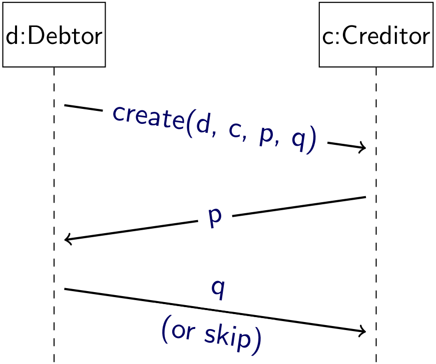
Operationalizing Commitments: Discharge First; Optional Detach

How about this?



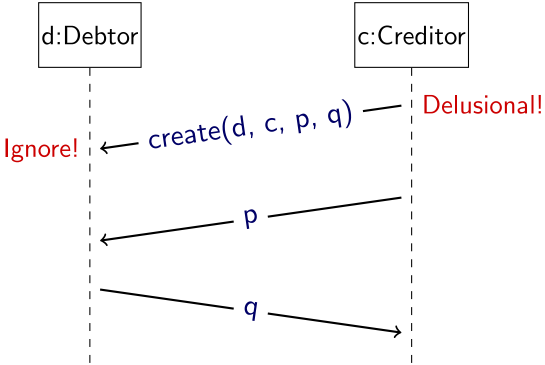
Operationalizing Commitments: Detach First; Optional Discharge

How about this?



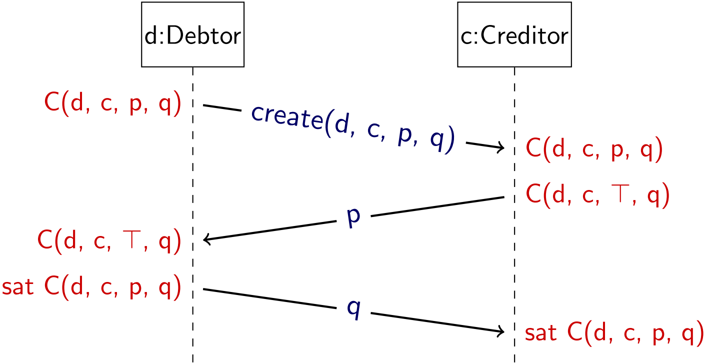
Operationalizing Commitments: Creation by Creditor

C(debtor, creditor, antecedent, consequent)



Operationalizing Commitments: Strengthening by Creditor

C(debtor, creditor, antecedent, consequent)



What is a Sociotechnical System?

A microsociety

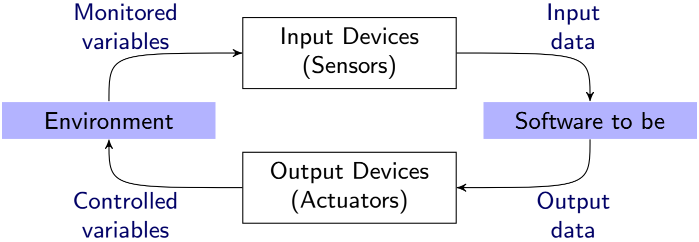
- Social entities, i.e.,
  - Autonomous principals
- Interacting with one another
  - Through and about technical entities
- *Norms*: Standards of correct behavior with which to judge self and others
- Must respect autonomy: a principal may apply any policy to decide its actions
  - May violate norms

Sociotechnical Systems in Current AI and SE

- An STS
  - Characterizes the organizational context of agents
  - Represents the problem world, which must be analyzed to derive requirements specifications for software
- Specifically,
  - Deemphasize autonomy and norms
  - Model devices and software on par with social entities

Traditional AI and SE: Single-Agent

Two “participants”: agent (software to be) and environment



- Methodologically: Refine requirements into a software specification
- Compile out the social parts

Specifying Sociotechnical Systems

Why restrict ourselves to technical entities?

- What is the nature of a specification for STSs?
- How may we derive such a specification from requirements?
- How do STS specifications relate to agent specifications?
- What do we gain from specifying an STS and then the agents instead of specifying an agent (or even multiple agents) directly?

Regimentation: Preventing Violations of Norms

- In an automobile insurance STS, suppose we have this requirement:
  - Any replacement parts ordered by a mechanic that are priced above $500 must be approved by the insurer
- Traditional approaches rely on technical solutions: produce agent specifications to enforce some requirement
  - Specify a agent that disables orders for parts over $500 unless the insurer approves them first
- Regimented solutions curtail deviations and innovations
  - What happens if a mechanic wants to urgently place the order?

Regulation: Stating a Norm but Allowing its Violation

Express the previous requirement as a norm

- $\{\mathsf{C}(\textsc{Mechanic}, \textsc{Insurer}, \mathsf{price}(\mathit{item}) > 500, \mathsf{getApproval}(\mathit{item})$ precedes $\mathsf{order}(\mathit{item}))\}$
- Benefits
  - Captures the essence of the requirement
  - Enables determining accountability and compliance
  - Enables a principal to design agent according to own requirements, perhaps one that is designed to be noncompliant in some situations

Characterizing Requirements Engineering

- "Machine" orientation (Zave and Jackson)

$$A, M \vdash R$$

- Sociotechnical orientation

$$A, Ag, S_P \vdash R$$

  - Collapses to Z&J when Ag is a singleton and, therefore, S_P is a null protocol
  - Principals via their agents are runtime participants in the system
  - An STS is a multiagent system, not just an environment and agent

Engineering Sociotechnical Systems

- Idea that STS specifications are not agent specifications but protocol specifications
  - Expanded scope of software engineering for AI
  - Specification of protocols is independent from the specification of principals' agents
    - Enables principals to act flexibly and in innovative ways
- Social refinement calls for
  - Methods for deriving protocol specifications from STS stakeholder requirements
  - Logic for relating satisfaction of requirements to protocol specifications and assumptions

Directions

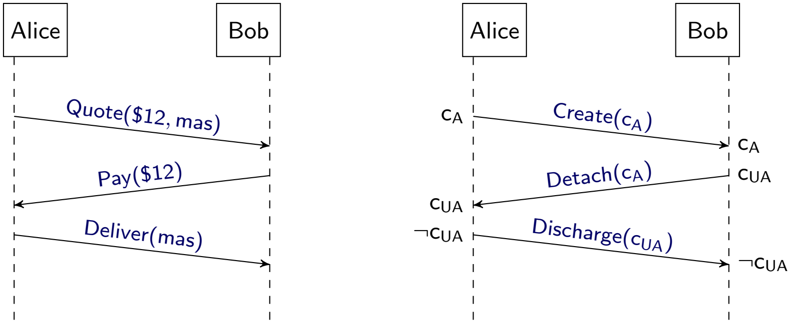
Protos: An anagram of Tropos

- Handling conflicting requirements, potentially by satisficing rather than satisfying requirements
- Extensions to other kinds of norms, e.g., authorization, power
- Formalizing other useful reductions, e.g., for capturing delegation
- Devising methodologies and tools
  - To give guidance to STS stakeholders on how to apply the reductions
  - To give guidance to agent designers

Commitment Progression via Messaging

Two views of the same enactment

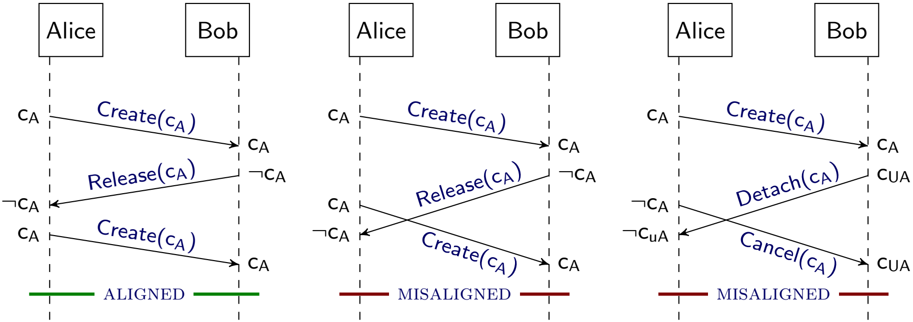
- C_A is C(Alice, Bob, paid(\$12), deliver(jam))
- C_{UA} is C(Alice, Bob, $\top$, deliver(jam))



Asynchrony May Cause Misalignment

Does Bob infer a commitment that Alice doesn't?

- c_A is $C(\text{paid}(\$12), \text{deliver}(\text{jam}))$
- c_{UA} is $C(\top, \text{deliver}(\text{jam}))$



Commitment-Level Interoperability

Alignment with respect to a creditor's expectations

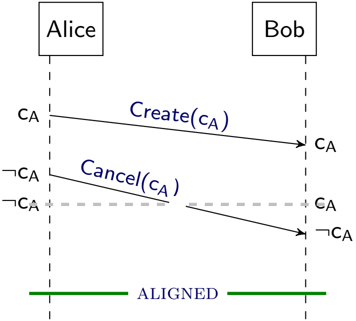
- Operational interoperability: Liveness of composed system
- Commitment alignment
  - Vector of observations $O = [O_1 \ldots O_n]$:, one per agent
  - O is causally valid: if w receives a message from z then z must have sent that message to z
  - O is aligned wrt $\mathsf{C}(x, y, r, u)$ iff

$$O_y \vdash \mathsf{C}(x, y, r, u) \Rightarrow O_x \vdash \mathsf{C}(x, y, r, u)$$

  - Nota Bene: this is asymmetric
- Interoperability: Either O is aligned wrt c or no matter what happens, O will extend to an O' that is aligned wrt c

Alignment When the Relevant Messages have Landed

- c_A is $C(paid(\$12), deliver(jam))$
- c_{UA} is $C(\top, deliver(jam))$



Challenge: Guarantee Alignment

By appropriately constraining an agent's local computations

- Formalize updates by messages indicating creation, cancelation, release, detach, and discharge
- Formalize information propagation constraints
  - Notify creditors of discharges
  - Notify debtors of detaches
- Taking into account commitment reasoning

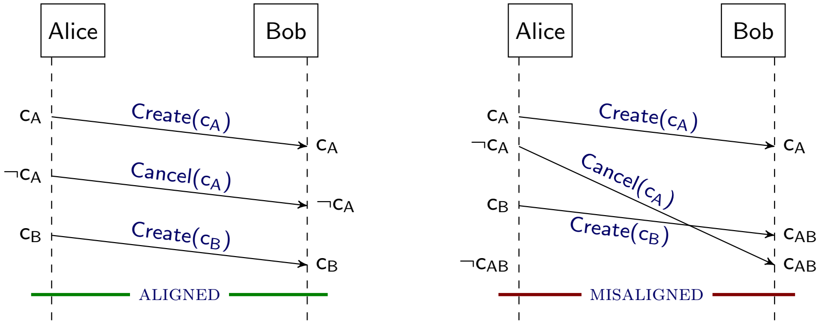
$$\text{Detach. } \mathsf{C}(r \wedge s, u) \wedge r \rightarrow \mathsf{C}(s, u)$$
$$\text{L-Disjoin. } \mathsf{C}(r, u) \wedge \mathsf{C}(s, u) \rightarrow \mathsf{C}(r \vee s, u)$$
$$\text{R-Conjoin. } \mathsf{C}(r, u) \wedge \mathsf{C}(r, v) \rightarrow \mathsf{C}(r, u \wedge v)$$

- Prove: Interoperability

Challenges of nonFIFO Message Delivery

A weak commitment cannot be canceled when a strong commitment holds

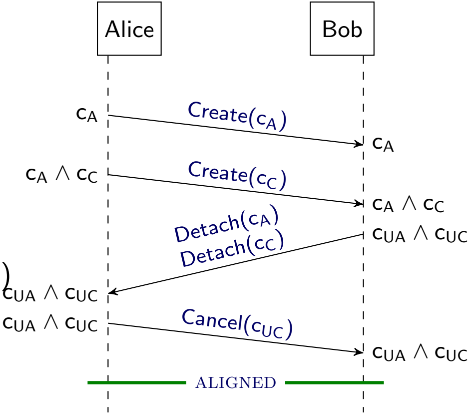
- C_A is C(paid(\$12), deliver(jam))
- C_B is C(paid(\$12), deliver(tofu))
- C_{AB} is C(paid(\$12), deliver(jam) $\wedge$ deliver(tofu))
- $C_A \wedge C_B \vdash C_{AB}$



Transaction-Related Challenges

Aligned but unrealistic outcomes due to the lack of transactions

- C_A is $\mathsf{C}(\text{paid}(\$12), \text{deliver}(\text{jam}))$
- C_C is $\mathsf{C}(\text{paid}(\$15), \text{deliver}(\text{tofu}))$
- C_{UC} is $\mathsf{C}(\top, \text{deliver}(\text{tofu}))$
- C_{UAC} is $\mathsf{C}(\top, \text{deliver}(\text{jam}) \wedge \text{deliver}(\text{tofu}))$
- $C_{UA} \wedge C_{UC} \vdash C_{UAC}$



- Cancel strongest possible (unconditional) version of any commitment
- Atomicity: Cancel an entire transaction, not parts